ALISSON THOMAZ AQUINO

CARLA GABRIELLE MOTIN

DANIEL PITANGA DOS SANTOS

LUCAS BUENO DOS SANTOS

LUCAS GONÇALVES DA COSTA SILVA

BATTLESTADIUM: UM JOGO DE ESTRATÉGIA E AÇÃO

Trabalho de Conclusão de Curso apresentado ao Curso de Tecnologia em Análise e Desenvolvimento de Sistemas da Universidade Federal do Paraná como requisito à obtenção do título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Orientador: Prof. Msc. Pedro R. Torres Júnior

CURITIBA

2013

RESUMO

Este trabalho apresenta a criação de um jogo de estratégia e ação, que pode ser disputado por diversos jogadores em diferentes plataformas. Construído na linguagem de programação Java, portanto possuindo a capacidade de ser portátil a qualquer sistema operacional que contenha uma *Java Virtual Machine* (JVM) instalada e operante. O jogo foi nominado de BattleStadium e tem a possibilidade de ser disputado localmente no equipamento, ou em uma conexão por rede de acesso local com outros jogadores. O jogo tem a sua estratégia constituída em buscar os melhores caminhos em um labirinto, bloqueados momentaneamente, que poderão ser desbloqueados pelo próprio jogador ao inserir um objeto denominado de bomba, que permitirá o progresso do seu personagem. A ação é determinada quando os jogadores aproximam-se entre si, elevando o grau de dinamismo e emoção no jogo, pois o objetivo é a eliminação do oponente. Durante a abertura dos caminhos, o personagem poderá encontrar objetos que o ajudarão a concluir seus objetivos no contexto da disputa. Após as partidas, o usuário poderá consultar uma pontuação e classificação geral dos jogadores cadastrados, e verá assim o seu desempenho, além do jogo possuir um local para a exibição de prêmios alcançados por intermédio dos resultados dos jogadores, assemelhando-se a um quadro de troféus.

Palavras Chave: Bomberman, jogo, multijogador, multiplataforma.

ABSTRACT

This paper presents the creation of an action and strategy game that can be played by several players on different platforms. Built in the Java programming language, it has the ability to be portable to any operational system that includes a Java Virtual Machine (JVM) installed and running. Was given to the game the name BattleStadium and it has the ability to be played locally on the device or in a connection for local access network with other players. The game has its strategy consists in seeking the best paths in a maze, blocked momentarily, which can be unlocked by the player himself when inserting an object called a bomb that will make your character progress in the map. The action is determined when players are close to each other, increasing the degree of dynamism and excitement in the game, when the goal is to eliminate the opponent. During the opening of the path in the map, the character can find objects that will help you complete your objectives in the context of the dispute. After the matches, the user can query a score and overall rating of registered players, and so you will see your performance, and the game has a place for the display of awards achieved by means of the results of the players, resembling a box trophies.

Key Words: Bomberman, game, multiplayer, multiplataform.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

Abragames - Associação Brasileira de Desenvolvedores de Jogos Digitais

BSP - *Board Support Package*

DD – *Design Document*

Dejus - Departamento de Justiça, Classificação, Títulos e Qualificação

ESRB – Entertainment *Software Rating Board*

GDD - *Game Design Document*

GPL - *General Public License*

IDE – *Integrated Development Environment*

IP – *Internet Protocol*

JAR – *Java ARchive*

JVM – *Java Virtual Machine*

LAN – *Local Area Network*

MIT - *Massachusetts Institute of Technology*

MMORPG - *Massive Multiplayer Role Playing*

NES – *Nintendo Entertainment Systems*

NPCs – *Non-Playable Characters*

RPG – *Role-Playing Game*

RUP - *Rational Unified Process*

SFX - *Special Effects*

SGBD – Sistema Gerenciador de Banco de Dados

SNES – *Super Nintendo Entertainment Systems*

UML – *Unified Modeling Language*

USB – *Universal Serial Bus*

WBS - *Work Breakdown Structure*

XML - *eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

No momento presente, constata-se um crescente interesse de todo um público na indústria de jogos eletrônicos, sendo que a cada passo de inovação nessa área possibilita a abertura de grandes ramificações dos gêneros. Um gênero que tem se destacado nos últimos tempos são os jogos de estilo retro, ou seja, jogos modernos, mas com uma estética que remete aos jogos antigos, de gráficos de oito a dezesseis *bits*. Na década de 90 e início do ano 2000, vemos alguns grandes títulos de jogos que se perderam no tempo, com algumas empresas não tendo um retorno preciso do seu público e até mesmo problemas de financiamentos, alguns jogos acabaram caindo no esquecimento. (Abragames, 2008).

Para a confecção deste projeto foi necessário pesquisar ferramentas que pudessem proporcionar recursos suficientes para modelagem do sistema, das imagens, processamento gráfico de jogos, mapeamento das telas, execução de arquivos de som e efeitos sonoros, serialização de objetos e envio de informações via rede local - *Local Area Network* (LAN). Através de tais pesquisas, revelou-se como melhor opção utilizar a biblioteca Slick-Util, para o desenvolvimento de jogos de segunda dimensão em linguagem Java, assim como também a biblioteca X-Stream, para serialização de objetos e exportação via rede em formato *eXtensible Markup Language* (XML), para viabilizar a implementação não apenas de um jogo no modo *offline*, mas também com a possibilidade de partidas via LAN.

Neste projeto, o objetivo foi ressuscitar um jogo chamado “Bomberman”, que só estava disponível em consoles antigos que foram descontinuados. Desse modo, foi notada a importância em desenvolver o jogo em Java, uma linguagem multiplataforma, que pudesse tornar o jogo com portabilidade entre plataformas computacionais, utilizando metodologias de desenvolvimento e as bibliotecas citadas anteriormente. Foi desenvolvido, portanto, um jogo de ação e estratégia, contendo modo *offline* e um modo multijogadores para resgatar o estilo antigo do consagrado jogo, mas inserindo novos mapas e personagens

para relembrar a jogabilidade que encantou a tantos na década de 90 e início de 2000.

1.1 OBJETIVOS DO PROJETO

O objetivo é desenvolver um jogo eletrônico no estilo de ação e estratégia, implementando modos de jogo tanto *offline* como um modo multijogador através de uma rede LAN. O cliente deste trabalho é a própria equipe, sendo possível posteriormente ser aberto ao público. Os jogos antigos sempre eram publicados para diversos consoles, tais como *Nintendo Entertainment Systems* (NES) de oito bits gráficos e *Super Nintendo Entertainment Systems* (SNES) de dezesseis *bits*, no entanto, ainda careciam de versões compatíveis para computadores.

Com tais informações, a equipe viu a oportunidade do desenvolvimento de um jogo nessa plataforma, com um objetivo de resgatar a essência do jogo e transformá-lo de uma maneira mais interativa, como por exemplo, suporte a controle de teclado, controles USB, modo multijogadores tanto *offline* como conectado via LAN.

A implementação do jogo deve conter um mapa, aonde os jogadores possam movimentar-se e colocar bombas estrategicamente com o objetivo de derrotar o seu oponente. O jogo deve também ter suporte não apenas ao teclado, mas a dispositivos USB, segundo a vontade do jogador para controlar o seu personagem. Para a implementação do jogo via LAN, deve conter uma configuração do IP ao se conectar, para que ambos os jogadores ligados nessa rede possam começar o jogo, trocando informações entre si sobre as informações pertinentes aos personagens. Também deverá estar implementado um servidor de aplicação que consiga gerenciar o cadastro de usuários para fins de contabilização da pontuação dos personagens durante o jogo em rede.

Para que tais objetivos sejam alcançados, uma lista de características específicas explica o que dever ser implementado:

- Cadastro de usuário;
- Identificação de usuário e pontuação;
- Implementação dos modos de jogo, *online* e *offline*;
- Implementação de um servidor de aplicação para gerenciamento de contas;
- Desenvolvimento de interfaces de configuração de sistema;
- Desenvolvimento de interfaces de gerenciamento de pontuação do usuário;

1.2 ORGANIZAÇÃO DO TRABALHO

O capítulo dois descreve as concepções e fundamentações necessárias para uma melhor compreensão do projeto, como a história, conceitos, programação, design e elementos de jogos eletrônicos. No capítulo três contem as metodologias utilizadas para a organização do projeto, com modelos, planos de atividades, diagramas, materiais utilizados para o desenvolvimento do projeto. O capítulo quatro trata-se de como o projeto foi desenvolvido, apresentando uma análise do jogo em seu desenvolvimento, protótipos e funcionalidades das telas. O capítulo cinco é uma apresentação geral do jogo, possuindo uma visão geral do seu conteúdo, com as funcionalidades e um guia de como o jogo poderá ser instalado. No capítulo seis possui as considerações finais do desenvolvimento do projeto, contendo um resumo dos resultados e propostas futuras de trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

O objetivo deste capítulo é apresentar sobre o desenvolvimento de jogos eletrônicos, abordando sobre a história dos jogos, as diferentes tecnologias existentes para o desenvolvimento e que servirão como base na implementação deste projeto.

2.1 JOGOS ELETRÔNICOS

Um jogo eletrônico é um programa de computador que possui grande interatividade com quem o utiliza. Os jogos eletrônicos, em geral, proporcionam diretamente entretenimento e diversão, podendo também auxiliar no aprendizado, como é o caso dos jogos educacionais. Segundo Chris Crawford, *designer* de jogos e escritor norte americano, um jogo é uma atividade que possui objetivos definidos e que permite a interação do jogador com o próprio jogo ou com outros jogadores. (ARANHA, 2004).

O primeiro jogo foi criado por Willy Higinbotham em 1958 e recebeu o nome de *Tennis Programing*, também conhecido como *Tennis for Two*. Em 1961, Steve Russell criou o game *Space War*, juntamente com estudantes do *Massachusetts Institute of Technology* (MIT). Como os mainframes, grandes computadores que servem para processar um grande volume de informações em grande velocidade, eram muito caros e ocupavam muito espaço, em 1971, Nolan Bushnell, criou a *Computer Space*, uma máquina só para jogar *Space War*. Mais tarde Bushnell funda a empresa Atari, empresa pioneira em jogos eletrônicos. Em 1976, foi lançado no mercado o *Fairchild Channel F,* o primeiro videogame programável, que permitia congelar o jogo, alterar o tempo e a velocidade. Um ano depois, a empresa *Midway s* lança o jogo *Gunfight*, o primeiro a usar microprocessadores ao invés de um emaranhado de circuitos. Em 1978, dois lançamentos aumentaram ainda mais a popularidade dos games: *Football*, da empresa Atari, e o lendário jogo *Space Invaders*, importado pela empresa *Midway* e desenvolvido pela empresa Tait. Um ano

depois surge o primeiro jogo com gráficos vetoriais, onde os objetos eram formados por linhas, como se fossem um esqueleto de um modelo 3D. Em 1980, a empresa Namco lança o famoso jogo *Pac Man*. Um ano depois, a empresa *Nintendo* obteve destaque com o *Donkey Kong*. Nesta época também foi criado o primeiro jogo de computador a incluir som digital, o *Castle of Wolfenstein*. (ARANHA, 2004).

A evolução dos jogos desencadeou-se da evolução dos computadores. Atualmente, com recursos mais aprimorados, os jogos simulam a realidade e promovem cada vez mais a interatividade. Segundo Diego Borges, em 2013, o game *BioShock Infinite* impressionou com seus cenários cheios de vida. Outros exemplos de jogos da atualidade são *Company of Heroes 2, Gears of War, Resident Evil, GTA* entre outros. (ARANHA, 2004).

A respeito do mercado de jogos, segundo a Associação Brasileira de Desenvolvedores de Jogos Digitais (Abragames), atualmente o Brasil é um dos maiores consumidores de jogos eletrônicos no mundo. Na área de desenvolvimento de games, as funções mais procuradas são *designers* e programadores de jogos para redes sociais e plataformas móveis, como celulares e computadores portáteis.

2.1.1 Jogos Eletrônicos em Rede

Os jogos eletrônicos em rede foram criados a fim de aumentar a competividade e a interação entre os jogadores. Os jogos online tem a vantagem de não precisarem ser instalados, pois podem ser acessados por um browser.

A arquitetura cliente-servidor é quando uma pessoa abre a aplicação do seu jogo (cliente) e faz o *login*, é enviada uma requisição ao servidor do jogo. Esse servidor acessa um IP que possui um aplicativo chamado *listener*, também conhecido como *server*. O *listener* através de uma configuração TCP/IP, “escuta” requisições em uma porta. Quando chega uma requisição, o

listener interpreta e devolve uma resposta a quem fez a requisição. (TANENBAUM, 2004).

Um exemplo de jogo eletrônico online é o Massive Multiplayer Role Playing (MMORPG). Em um MMORPG, é possível montar um time com vários jogadores. O jogo “World of Warcraft” é um exemplo de MMORPG. Para rodar um MMORPG, basicamente é necessário um cliente e um servidor. O cliente é o programa que roda no computador e o servidor é a máquina na qual você se conecta quando vai jogar. O cliente combina a tela de jogo e a janela usada para ver o jogo. Quase tudo sobre o jogo fica em arquivos e base de dados do computador. O servidor possui várias responsabilidades. Ele compara a posição do jogador em relação aos outros, os monstros do jogo (mobs) e os personagens controlados pelo próprio computador (NPCs), notifica o cliente quando está sendo atacado, e muitos outros. Na verdade, os MMORPGs também necessitam de outros servidores como um servidor de autenticação onde os jogadores se conectem e acessem o jogo, um servidor de bate-papo, para possibilitar aos jogadores comunicação e um servidor *web*, no qual os jogadores possam acessar informações sobre suas contas.

2.2 DESIGN DE JOGOS ELETRÔNICOS

Os jogos eletrônicos devem estar baseados em um *design* específico ao objetivo proposto inicialmente do projeto. O *designer* deverá descrever as funções e regras que farão o jogo único e especial, e também deve solucionar problemas de áreas vazias ou ignoradas do jogo. Por isto os *designers* precisam ter liberdade para decidir, livres de tecnologias ou limitações, antes guiando-se pelo seu gênio, inovação e visão do jogo como um todo.

O *design* do jogo pode ser dividido por uma equipe multidisciplinar que ajudarão um em cada área, no desenvolvimento do jogo. Os artistas e animadores provem o trabalho da arte e animações. Os programadores irão trabalhar em conjunto com os artistas para que eles possam ver suas artes em tempo real de jogo. Os produtores devem ter certeza que os programadores

provêm uma versão atual do jogo para vendas, relações públicas, e para o time de marketing, e também os vários relatórios sobre as últimas versões de jogo. “Estes relatórios descrevem como o modo de jogo é realizado, características especiais, requisitos de hardware [...] e contem fotos de tela para fins de propagandas, revisões em revistas”. (PEDERSEN, 2003, p. 4). Os testadores *Quality Assurance* (QA) relatam problemas para o produtor, os problemas podem ser divididos em maiores (travamentos, função ou ação que não funcionam), e menores (textos com erros, caracteres movendo-se muito lentamente ou rapidamente), falhas (problemas de som ou gráfico), melhoramentos (adicionar uma nova característica, melhorar a interação do personagem e seu comportamento), e inconsistência de multiplataformas (versão de computador e versão de console). (PEDERSEN, 2003).

Um princípio básico no *design* de games é a simplicidade na criação, pois todo o aspecto do produto deve ser óbvio e fácil de entender. Um exemplo é a combinação de teclas, pois é mais fácil que o usuário consiga acessar muitas opções com apenas dois botões, do que com uma combinação difícil de entender e aplicar. Manter a interface de *design* simples é um dos princípios que devem estar sempre à vista de quem deseja desenvolver um jogo com qualidade. (PEDERSEN, 2003).

Os jogos devem conter maneiras diversificadas de como ele poderá terminar. Eles também precisam ter um objetivo, por exemplo, em jogos de esportes o objetivo é ganhar na pontuação de seus adversários, e assim por diante. Algo importante também que deve ser salientado, é que o jogo deve ter a possibilidade do jogador conseguir a vitória. Em um modo de multijogadores, sejam dados a cada jogador possibilidades iguais como forças e fraquezas ao iniciar. Um bom jogo traz a cada jogador chances iguais para a vitória.

2.2.1 Design de interface

O *design* de interface abriga um conjunto amplo de responsabilidades quando se trata de jogos eletrônicos, que são as seguintes: modelagem de

interação do usuário com o produto, criação de protótipos de jogabilidade, implementação de diversos recursos de *design*, e tudo isso dentro do requisito principal, que é deixar o jogo mais atraente e interessante em diversos aspectos.

Existem algumas preocupações nesta área, tais como o limite a ser estabelecido no uso de tecnologias. Esse limite deve ser bem definido a fim de que não se tome nenhuma direção errada. Determinados projetos, por exemplo, partem para do princípio de que a sensação de imersão em recursos realísticos seja um objetivo essencial para o público, porém em alguns casos, não deve ser supervalorizados, de tal modo que exista a consciência de que o usuário está jogando e não vivenciando algo real, fato essencial para o melhor aproveitamento do que se chama de entretenimento. Deste modo, podemos dizer que o uso indevido de algumas técnicas pode causar o efeito contrário do que se espera, como o incomodo causado por um jogo com interface contendo realidade excessiva. (PEDERSEN, 2003).

Para alcançar os objetivos de interface, pesquisas de qualidade com o público que se espera atingir poderão ser feitas, dando a possibilidade de identificar qual linguagem gráfica, estilo de interação, e outros recursos são preferidos.

2.2.2 Design de Áudio

Um fato pertinente em relação ao áudio nos diversos tipos de jogos é possuir uma trilha sonora musical que é tocada em diversificados momentos. Desde a tela inicial a escolha da música deve possuir concordância com a temática, podendo trazer pontos positivos dependendo da escolha. Uma escolha certa pode simplesmente aliar o jogo com a trilha, que quando ouvida em outra ocasião, automaticamente vem à tona a lembrança do jogo, causando até mesmo um sentimento de nostalgia. Isso é o que acontece como, por exemplo, no jogo FIFA 1998, com a música *Song 2 da* banda *Blur*. Uma pesquisa de trilhas seria o ideal para uma boa definição nesta etapa. Na

programação o áudio é chamado entre as linhas de códigos e associados a um determinado evento e interação com usuário, por exemplo, quando o usuário estiver em piso de madeira, o som dos passos na madeira serão ativados dependendo do determinado movimento realizado pela interatividade.

Os efeitos sonoros preenchem a gama de recursos de um jogo, são usados sons artificiais ou gravados por sonoplastas e *designers* de áudio. As produções maiores investem uma grande quantia de dinheiro neste setor, que produz e cria cada barulho em um nível de gravação profissional, componente que exige grande custo e tempo. Porém faz grande diferença dando emoção a cada detalhe de cada etapa, em alguns casos intensifica o drama e o significado de imagens. Estes efeitos devem ser alinhados à interface, com o mesmo nível de qualidade. Também existe o outro lado, que indica erros na escolha do áudio. (PERUCIA E OUTROS, 2005).

O áudio não pode ser repetitivo ou causar uma poluição sonora para não ocasionar desinteresse por parte do usuário, por isso desde sons curtos até trilhas devem ser bem planejadas. A questão a ser respondida é: como se comportará a cabeça de quem se mantém jogando durante um grande período de tempo? O som está sendo muito cansativo? Existem momentos em que a ausência de áudio pode ser a melhor escolha. Ai entra outro fator importante neste caso que é a flexibilidade, que possibilita o usuário a baixar ou diminuir o volume, podendo escolher a intensidade dos sons ambientes tanto quanto trilha sonora.

2.3 PROGRAMAÇÃO DE JOGOS ELETRÔNICOS

A programação de jogos eletrônicos é um conjunto de etapas cuja sua totalidade permite o desenvolvimento de uma maneira eficiente, com garantia de um produto de qualidade no fim de todo o processo.

Muitos dos profissionais na área de desenvolvimento de jogos eletrônicos começam o processo de desenvolvimento a partir de um projeto, ou

usando o termo mais comum, pelo *designer*. Utilizando como exemplo a seguinte definição de *designer* é:

> [...] é o que determina a jogabilidade, as escolhas que o jogador terá dentro do mundo do jogo e as ramificações que suas escolhas vão ter no resto do jogo. Inclui o que faz o jogador vencer ou perder, como ele vai controlar o jogo, as informações que o jogador deverá receber. (PERUCIA E OUTROS, 2005, p.30).

Entretanto, em alguns casos vemos que um jogo nasce de um simples protótipo, organizando diferentes ideias e elementos em uma fase de pré-produção, para ter uma base de que características o jogo pode receber, ou também, o que o jogador, quando em controle de seu personagem, quais ações ele pode fazer e como isso afeta o jogo.

De qualquer maneira, a criação de um projeto é o que irá definir como o jogo deverá ser desenvolvido, incluindo detalhes não apenas a parte visual do jogo, mas levar em consideração toda a parte da interação entre homem-máquina, para garantir uma maior imersão do jogador. Esse projeto é criado em conjunto com toda a equipe responsável pelo desenvolvimento. Muitos dos requisitos retirados dos jogos se dão através da experiência, que nada mais é, ver o que a atual indústria tem conseguido produzir, mas também como da pesquisa de novas formas de se apresentar um jogo e a busca de elementos que aumentem o interesse do consumidor quando vê o produto. Todos esses detalhes geram um documento chamado *“Game Design Document”*, que se trata de um documento que consegue sintetizar todos os elementos anteriormente debatidos. Podemos ter uma ideia mais clara a partir da explicação sobre *Game Design Document* (GDD):

> O *Design Document* (DD) é como um script de um filme, que informa todos os detalhes do jogo. Escrever um DD é extremamente trabalhoso e exaustivo, pois se deve detalhar tudo que ocorrerá no jogo. Todavia é muito útil para repensar decisões já tomadas, validar alguns conceitos, suprimir algumas regras e adicionar outras. Você poderá até mesmo testar algumas ideias economizando tempo precioso para o projeto. Durante a escrita do DD, poderá visualizar com antecedência o jogo. O universo deve ser coerente, para que o jogador realmente entre no mundo que você estará propondo. (PERUCIA E OUTROS, 2005, p.33).

Feita a documentação, começa a etapa de produção, onde os programadores começam a trabalhar para desenvolver todo o código necessário previsto no documento de *design* criado. E é nessa etapa que as ideias do documento são validadas à luz das possibilidades que a linguagem proporciona, ou seja, validar se aquilo que está previsto no documento é possível ou não desenvolver, se será necessário ajustes ou até mesmo modificação do projeto, seja modificação dos requisitos, ferramentas, linguagem de programação, entre outros.

Sobre as ferramentas, diversas são as linguagens de programação, dentre elas, as mais comuns que vemos são: *C, C++, Java, ActionScript, Flash, C#, XNA,* entre outros. Cada linguagem contem suas próprias características e limitações, por isso, no estudo do desenvolvimento de um jogo, é necessário conhece-las para ter certeza de que os requisitos consigam ser desenvolvidos sem maior problema.

Outro aspecto importante é a comunicação entre as diversas equipes, como é comumente visto em grandes empresas, os programadores não só devem se comunicar com os projetistas, mas como todo o pessoal responsável pela arte do jogo, para obter detalhes pertinentes a personagens, cenários, itens, inimigos, entre outros.

Chega uma etapa em que já se tem boa parte do jogo desenvolvido, e é aonde a fase de testes começa com o objetivo de encontrar erros e bugs do jogo que muitas vezes não podem ser previsto durante a fase de produção. Não apenas isso, mas para verificar a qualidade do jogo que está desenvolvido, verificar se os requisitos que estão contidos no documento de game *design* estão sido cumpridos. Em projetos de alto custo, versões Alpha do jogo são lançados, muitas vezes para saber o *feedback* dos jogadores de o que acham do jogo até então e o que poderia ser incluso ou retirado. Já em projetos de baixo custo, não se tem uma versão de teste até que haja uma versão candidata pronta pra isso.

Após tudo isso, na fase de quase término de todo o processo de desenvolvimento, é onde ocorre o “polimento” do jogo. Programadores corrigem os erros e bugs encontrados durante a fase de testes. É comum

lançar uma versão *Beta* do jogo, que pode conter boa parte de todos os requisitos do jogo, apenas alguns erros e alguns módulos faltando. Isso também é feito para teste de tolerância de stress para os servidores de jogos, ou seja, conseguir medir qual é a media de acessos o jogo terá quando for publicado. Todo esse processo requer um acompanhamento da programação, que está inclusa na etapa de produção. No entanto, a programação é necessária em todas as fases do processo, exceto na fase dos conceitos principais do jogo, na fase mais inicial. E é ela, a programação, quem cuida da parte essencial do jogo, que é o seu motor.

2.3.1 Motor de jogo

O motor de jogo (ou somente *engine*) é um programa de computador ou um conjunto de bibliotecas responsáveis por simplificar funções do desenvolvimento do jogo, como “renderização” física e entrada de dados, dando liberdade para o desenvolvedor focar nos detalhes que tornam seu jogo singular.

Um motor de jogo possui subsistemas que estão presentes na maioria dos jogos, como os descritos a seguir:

- **Núcleo gráfico:** Transformam o modelo matemático do estado atual do jogo em uma visualização para o usuário. Essas funções controlam os elementos gráficos na tela como renderização 2D ou 3D, animação, texturas, fontes e câmera.
- **Núcleo de entrada:** Responsável por identificar os eventos de entrada de dados, capturando informações enviadas através de dispositivos tais como mouse, teclado e *joystick*.
- **Núcleo de som:** Responsável por executar os sons a partir de eventos ocorridos no jogo e pré-definido em suas respectivas funções.
- **Núcleo de sistema:** Responsável por gerenciar a criação de janelas, gerenciamento do estado e dos processos do jogo. Sua organização varia conforme o projeto do jogo.

Com a utilização de motores de jogos, podemos observar algumas vantagens visíveis, tais como: simplificar o desenvolvimento do jogo, utilizando chamadas à biblioteca que executam funções que facilitam ao desenvolvedor; facilita a portabilidade do jogo entre bibliotecas gráficas e até mesmo plataformas, onde as modificações irão acontecer somente na chamada das bibliotecas gráficas e não no jogo e sua estrutura; organiza o gerenciamento do código, disponibilizando adicionar novas funções facilmente; abstratar o trabalho, evitando que o desenvolvedor se preocupe com detalhes em baixo nível como chamadas de funções para ler arquivos de som, imagem e recursos.

2.3.2 Ferramentas de Autoria

Ferramentas de autoria são recursos amigáveis, onde programadores ou não, possam desenvolver com rapidez um determinado conteúdo ou programa. Essas ferramentas também são ótimas para desenvolver habilidades como game *designer* ou mesmo algumas técnicas de inteligência artificial. *"Maker Neverwinter Nights – Aurora Toolset" e "Oblivion – TES construction set"* são exemplos de ferramentas de autoria.

Para desenvolver um jogo de computador, os programadores devem possuir facilidade também na parte artística, para que o jogo tenha um bom enredo, gráficos e sons. Uma opção para desenvolvimento de jogos são as ferramentas de autoria. A infraestrutura foi projetada para oferecer um serviço confiável e escalável para os consumidores que utilizam esta tecnologia.

2.4 ELEMENTOS DE JOGOS

Os elementos de jogos são partes integrantes que, pelo composto, fazem com que o jogo possua um dinamismo, e consiga aproximar-se da realidade, ou trazer o aspecto real ao jogador. Os elementos têm como objetivo

tornar o jogo mais prazeroso e emocionante. Cada elemento do jogo deve ser cuidadosamente analisado pelo *designer* de jogos e outros participantes da criação do jogo, que terão como função a combinação e harmonização de todos os elementos, tornando todas as partes envolvidas em uma só. Cada elemento deve ser desenvolvido para que, com a combinação de todos, faça com que o jogo funcione de forma dinâmica, os *sprites*, objetos e *sounds* são os principais elementos dos jogos.

2.4.1 Sprites

Sprites são objetos gráficos de duas ou três dimensões que se movem na tela do jogo. *Sprite* é uma palavra tradicionalmente usada na programação de jogos de computadores para as imagens que se movem ao redor do jogo em si. Normalmente os *sprites* são baseados em *bitmaps* (mapa de *bits*). Também há os *sprites* geométricos de objetos, eles possuem a característica de serem em escalas independentes e nítida, e leve em termos de uso de memória. Outra característica positiva é que os *sprites* geométricos podem ser facilmente rotacionados. (RUCKER, 2002).

O objetivo dos *sprites* é possibilitar que os usuários interajam com o cenário, possibilitando uma melhor jogabilidade e envolvimento no jogo. Uma caminhada pode ser representada através de uma alternância entre um *sprite*, como demonstra na FIGURA 1 que exibe o personagem pisando com o pé direito, seguido por outro *sprite* no qual o mesmo personagem esteja pisando com o pé esquerdo, dando uma sensação de que ele está caminhando.

FIGURA 1 – Exemplo de *Sprite*

2.4.2 Objetos

Os objetos são os blocos básicos de construção de um sistema. Os sistemas podem ser considerados como um grupo de peças interligadas chamados objetos, que podem ser físicos, resumo, ou ambas, dependendo da natureza do sistema. Exemplos de objetos em jogos podem ser peças individuais de jogo (como o "rei" ou "rainha" no xadrez), os conceitos do jogo (como o "banco" em *Monopoly*), os próprios jogadores, ou representações dos jogadores.

Áreas ou terrenos também podem ser considerados como objetos. Estes objetos interagem com outros objetos do jogo, formando uma interatividade e dinamismo na jogabilidade.

Os objetos são definidos pelas suas propriedades e comportamentos. Eles também são definidos por suas relações com outros objetos.

2.4.3 Sound

É uma parte importante do jogo, com os sons é possível sentir-se mais próximo a sensação que o jogo se assemelha à vida real. Para definir os sons dos jogos, são necessárias habilidades dos engenheiros de som, que tentarão criar um clima para o jogo com músicas de fundo, e também quando os personagens dos jogos fizerem alguma ação, ou quando entrar alguma ação no jogo aonde algum objetivo foi alcançado.

O *designer* de jogos deve fornecer uma lista que estará relacionada ao jogo para o engenheiro de som. São aspectos que devem ser analisados durante a criação: tempo estimado para a fase, parte, ou sessão do jogo, objetivos alcançados. O *designer* de jogos deve prover exemplos de músicas e efeitos sonos que ele gostaria que fosse aplicado ao jogo, isto de acordo com o tema e gênero do jogo em questão. Com esta lista e definições em mente, o engenheiro de som irá definir no processo de criação de sons, composições

originais, e efeitos sonoros que estarão ligadas ao que o *designer* de jogos deseja.

2.4.3.1 Biblioteca de Som

Na biblioteca de som são armazenados milhares de exemplos de sons já criados que os *designers* de jogos poderão usá-los para darem de modelos aos criadores de som, e estes poderão ter maiores exemplos para poderem criar novos tipos e alternarem entre opções. Estas bibliotecas também possibilitarão aos engenheiros de som, de acordo com a vontade do *designer* de jogos, escolherem diretamente sons para os efeitos sonoros e até mesmo para as músicas de fundo.

Existe uma biblioteca conhecida como *Special Effects* (SFX), que é uma biblioteca que possui uma grande quantidade de efeitos especiais para os jogos de todos os gêneros. São exemplos de SFX: Sound Ideas, Warner Brothers, Hanna Barbara, Disney, Universal Studios, 20th Century Fox, entre muitas outras. Existem muitas categorias de sons que poderão ser usadas, como: Sons gerais, sons de filmes, efeitos de desenhos e comédia, ficção científica, sons de terra e água, pássaros e animais, reação da platéia, sons ambiente, fala e pronúncia, entre diversos outros tipos. (Pedersen, 2003, p. 215 – 232).

2.5 GÊNERO DE JOGOS ELETRÔNICOS

O Gênero de um jogo eletrônico é uma classificação que visa definir o estilo de um jogo através de suas principais características e requisitos, como por exemplo, sua perspectiva, interação com o jogador, objetivos, entre outros.

Toda essa gama de gêneros foi criada pelas primeiras empresas a atuar nessa área, tentando classificar seus jogos de uma maneira similar ao que se classificam os filmes. Tais classificações são bastante variáveis, pois, através

de tantos de desenvolvimento e evoluções na área de desenvolvimento de jogos, novos gêneros e termos foram criados. A seguir uma sucinta explicação dos gêneros mais famosos.

2.5.1 Jogos de Ação

Jogos em tempo real que dependem dos reflexos do jogador para antigir certos objetivos. A FIGURA 2 demonstra o jogo *Super Mario*, um grande clássico dos jogos de ação, o jogador deve concluir as fases, eliminar diversos inimigos pelo caminho para conseguir chegar ao seu objetivo principal, que é resgatar a princesa no castelo do chefão.

A FIGURA 3, o jogo *Megaman X*, é um jogo que exige do jogador grande reflexo e atenção, aonde deverá desviar de ataques dos inimigos e conseguir passar por eles para chegar à fases com inimigos cada vez mais difíceis de matar.

FIGURA 2 – Jogos de ação: Super Mario

FIGURA 3 – Jogo de ação: Megaman X

2.5.2 Jogos de Aventura

Jogos com um passo mais lento e que é bastante focado em charadas e quebra-cabeças para que possa progredir no jogo. O jogo *King´s Quest VI*, na FIGURA 4, é um jogo aonde o jogador terá que passar por diversos obstáculos durante as fases, sempre com a dificuldade aumentando e a complexidade dos elementos também, tornando o jogo emocionante.

FIGURA 4 – Jogo de aventura: King’s Quest VI.

2.5.3 Jogos de RPG (Role-Playing Games)

Gênero que se originou do famoso jogo *Dungeons and Dragons*, uma aventura de “papel-e-caneta” aonde se cria um personagem e vai ganhando experiência das missões que ele deve fazer no jogo, adquirindo novas habilidades para conseguir completar o objetivo do jogo.

A FIGURA 5, que trata do jogo *Neverwinter Nights*, demonstra que o jogo de RPG deve ser jogado de forma que de acordo com os objetivos alcançados, o jogador terá mais forças e poderá enfrentar criaturas ou outros adversários, de forma mais eficiente.

FIGURA 5 – Jogo de RPG: Neverwinter Nights

2.5.4 Jogos de Estratégia

Jogos de estratégia, são jogos que dado um número x de recursos, tempo e uma situação, você deve tomar decisões para otimizar a usabilidade de seus recursos para atingir um objetivo. Na FIGURA 6 o jogo *Super Bomberman 5*, é um jogo de estratégia, aonde os jogadores precisam inserir bombas ao decorrer do trajeto para abrir caminhos e também com o objetivo de eliminar os adversários, e assim conseguir pontos. Na FIGURA 7, o jogo *Age of Empires II,* é um jogo de pura estratégia, aonde é necessário conseguir juntar recursos como ouro, pedra e madeira, para alcançar objetivos, como construir casas, moinhos, centros de treinamento para os soldados, entre outros, que farão o jogador ter maior poder militar, financeiro, e poderá assim eliminar o seu adversário ou alcançar objetivos específicos no jogo.

FIGURA 6 - Jogos de estratégia: Super Bomberman 5

FIGURA 7 - Age of Empires II

2.5.5 Jogos Online

Jogos *online* são jogos capazes de estabelecer uma partida entre vários jogadores através de uma forma de conexão em rede, às vezes através da Internet ou de uma conexão local *Peer-to-Peer* (P2P) via LAN. O jogo *World of Warcraft,* na FIGURA 8, é um grande clássico dos jogos *online*. Neste jogo, o objetivo é conquistar objetivos breves, para que possa conseguir alcançar mais recursos militares de ataque ou defesa, e também envolve uma parte estratégica, que poderá ser usada contra o adversário.

FIGURA 8 – Jogos *online*: *World of Warcraft*

2.6 CLASSIFICAÇÃO DE JOGOS ELETRÔNICOS

A classificação de jogos eletrônicos é um sistema que se aplica a jogos e aplicativos móveis, sejam eles via mídia ou conteúdo. Sua principal função é dividir jogos em categorias de acordo com o conteúdo que cada um apresenta. Tomando como referência o sistema de classificação da *Entertainment Software Rating Board* (ESRB), organização que analisa, decide e coloca as classificações etárias indicativas para jogos eletrônicos comercializados na América do Norte, temos as seguintes categorias na tabela a seguir.

Selo	Nome	Classificação	Descrição
RATING PENDING RP ESRB	Rating Pending	Em Análise	O produto está em análise para uma classificação final. O símbolo aparece prioritariamente em versões não finais

			de jogos.
	Early Childhood	Maiores de 3 anos	Contém conteúdo considerado próprio para crianças até 3 anos. Jogos nesta categoria não contêm qualquer material impróprio. Esses jogos são especificamente destinados a crianças e jovens e geralmente de natureza educativa.
	Everyone	Maiores de 6 anos ou livre	Contém conteúdo considerado impróprio para menores de 6 anos, antigamente, mas agora a classificação é representada livre para todas as idades. Títulos nesta categoria podem conter violência animada ou fantasiosa leve.
	Everyone 10+	Maiores de 10 anos	Contém conteúdo considerado impróprio para menores de 10 anos. Títulos nesta categoria podem conter violência animada ou fantasiosa moderada, linguagem agressiva leve, sangue animado e/ou mínimos temas sugestivos.
	Teen	Maiores de 13 anos	Contém conteúdo considerado impróprio para menores de 13 anos. Títulos nessa categoria podem conter violência, temas sugestivos, humor cruel, pouco sangue, jogos de azar e/ou uso moderado de linguagem forte.

	Mature 17+	Maiores de 17 anos	Contém conteúdo considerado impróprio para menores de 17 anos. Títulos com essa classificação contêm violência mais forte com sangue, consumo de drogas ou álcool referências sexuais e/ou linguagem obscena.
	Adults only 18+	Maiores de 18 anos	Contém conteúdo considerado impróprio para menores de 18 anos. Colocada em jogos adultos que contêm representações de sexo e nudez extrema e/ou violência incluindo sangue.
	Kids to Adults	Maiores de 6 anos (depreciada)	Contém conteúdo destinado a crianças maiores de 6 anos ou mais. Títulos nessa categoria podem conter pouca violência e alguma linguagem forte. Foi trocada pela Everyone (todas as idades) no começo de 1998.

TABELA 1 – Classificação de jogos internacionais.

No entanto, a classificação utilizada no Brasil para jogos eletrônicos é a mesma utilizada para programas de televisão e filmes. O Departamento de Justiça, Classificação, Títulos e Qualificação (Dejus) é responsável pela classificação no Brasil. Tal classificação começou em outubro de 2001 e foi estabelecido pelas Portarias nº 1.035 e 899. As faixas de classificação brasileiras são as seguintes, conforme a FIGURA 9:

- Livre
- 10 anos
- 12 anos
- 14 anos

- 16 anos
- 18 anos

FIGURA 9 – Classificação brasileira dos gêneros de jogos.

Quem classifica os jogos dentro dessas faixas e a própria Dejus, que recebe dos produtores e distribuidores jogos de demonstração, fotos, vídeos dos jogos para que possam fazer a análise. Os critérios pelos quais eles se baseiam para classificar um jogo são: sexo, violência e drogas. As cores nas classificações, como mostrado na FIGURA 9, facilitaram para que o público reconheça de forma facilidade qual é a classificação.

2.7 TRABALHOS RELACIONADOS

Após várias pesquisas para aprimorar na fundamentação deste trabalho, foram encontrados diversos trabalhos acadêmicos relacionados a jogos eletrônicos.

2.7.1 Projeto e Desenvolvimento de Jogos Computacionais

O trabalho de conclusão de curso (TCC), feito por Gean Alex Pereira, em 2006, sob o título “Projeto e Desenvolvimento de Jogos Computacionais”, publicado na UFPR, apresenta as metodologias que podem ser utilizadas para o desenvolvimento de jogos. Gean além de apresentar essas metodologias, apresentou o desenvolvimento do jogo *The Quest*, onde o jogador enfrenta vários desafios e armadilhas em um labirinto.

O jogo utiliza o motor de jogos *Studio* (3DGS), que possui uma versão gratuita para teste. Os cenários do jogo foram desenvolvidos utilizando um modelador de cenário do próprio 3DGS.

Dentre as metodologias existentes para o desenvolvimento, o autor utilizou a *Design*, que é específica para o desenvolvimento de jogos de computador.

2.7.2 Quake

O jogo Quake é um jogo em primeira pessoa (FPS), de tiro que foi originalmente lançado em 1996. O jogo foi desenvolvido pela Id Software, e as músicas compostas por Sonic Mayhem e Front Line Assembly.

O motor deste jogo foi lançado em 1996 e sua principal atração na época era a “renderização” dos gráficos em terceira dimensão em tempo real. E em 1999 a Id Software liberou o código deste motor para fins de desenvolvimento na licença *General Public License* (GPL). A liberação do código do motor do jogo pôde ocasionar um grande número de novos jogos. Alguns jogos estão atualmente no mercado de jogos, como o *Call of Duty: Modern Warfare 3, Medal of Honor, 007: Nightfire, Doom,* e as continuações do próprio jogo *Quake 2, 3 e 4*, entre outros que puderam usar o código do motor original de *Quake.* Da primeira versão de *Quake* ao *Quake* 4, houve uma mudança significativa nos gráficos, na jogabilidade e certos aperfeiçoamentos, as FIGURAS de 10 a 13 demonstram a evolução do jogo e como ele foi alterando de acordo com as melhorias.

FIGURA 10 – Jogo Quake (1996).

FIGURA 11 – Jogo Quake II (1997)

FIGURA 12 – Jogo Quake III (1999).

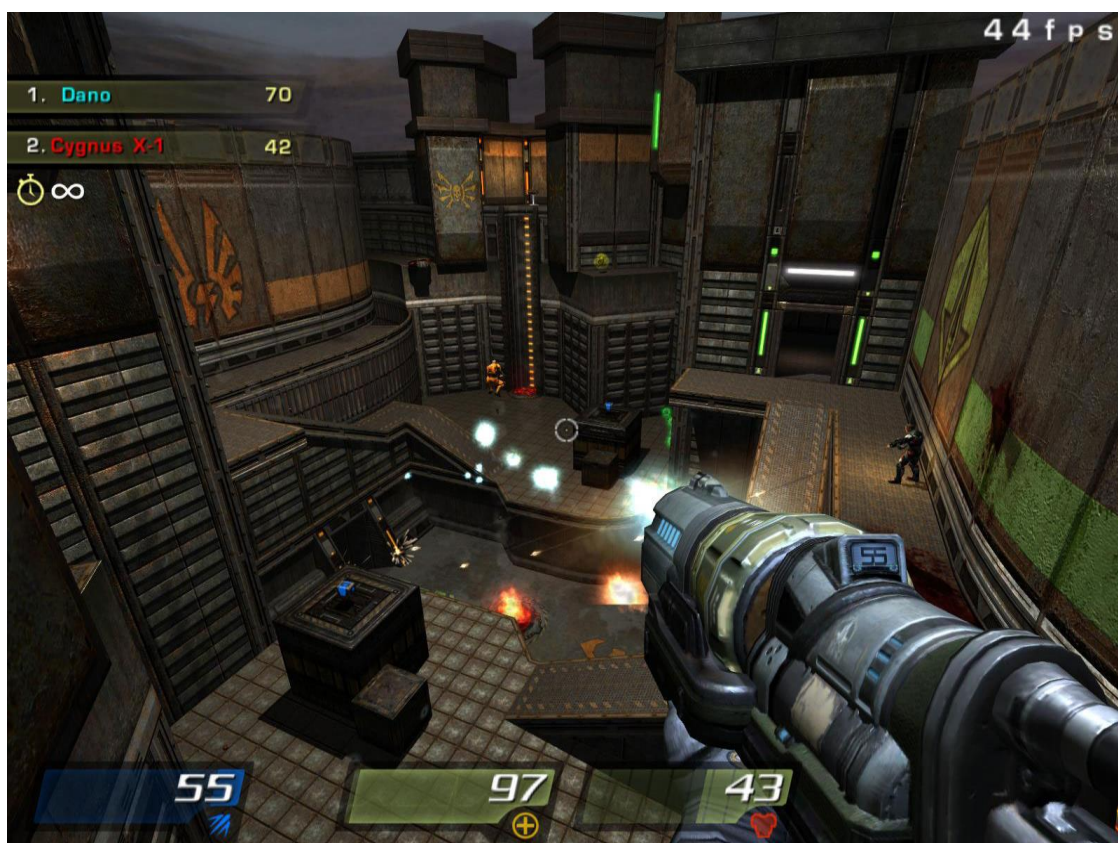



FIGURA 13 – Jogo Quake 4 (2005).

Atualmente o código de fonte do motor do jogo Quake III encontra-se disponível ao público na internet no site da GitHub. Segundo o relatório da Academic ADL Co-Lab, feito por Mohamed Eldawy, relatou que o código fonte está dividido em oito módulos menores, que são:

- **Botlib:** é a área ampla do jogo, onde os personagens inimigos irão usar a inteligência artificial para tomarem decisões.
- **Cgame:** é o cliente do jogo, onde todas as ações devem ser previamente aprovadas pelo servidor.
- ***Game*:** o servidor do jogo.
- **Q3_ui:** ui que significa *User Interface*, tem como objetivo a mostra dos menus e ações do jogador.
- ***Quake 3:*** é o arquivo executável do jogo.
- ***Renderer:*** responsável pelos gráficos de todas as coisas, ele vai conter o código *Board Support Package* (BSP), animações, e as texturas do jogo.
- ***Splines*:** é o módulo responsável pela geração de curvas nos gráficos e na interpolação entre elas.
- **UI:** é o módulo que trata da interação com o jogador.

3 METODOLOGIA

O objetivo deste capítulo é apresentar as metodologias e ferramentas que foram utilizadas no planejamento e desenvolvimento do jogo. Também serão apresentadas as responsabilidades atribuídas aos membros da equipe e como ocorreu o desenvolvimento do projeto.

3.1 MODELO DE PROCESSO DE ENGENHARIA DE SOFTWARE

O modelo utilizado neste projeto foi o RUP (*Rational Unified Process*), que fornece diretrizes para definir as tarefas e atribuir responsabilidades em um projeto. O RUP possui quatro fases: iniciação ou concepção, elaboração, construção e transição. Na iniciação basicamente define-se o escopo do projeto. Na elaboração é obtida uma visão abrangente do sistema, através da construção de protótipos e também é definida a arquitetura do sistema. Na construção, o foco está no desenvolvimento do sistema. Por fim, na transição, o produto é transferido ao usuário e o projeto é avaliado e pode ser concluído. Em todas as fases há o gerenciamento dos requisitos e dos recursos do projeto. Por isto o RUP é baseado no desenvolvimento iterativo, que por sua vez é mais flexível quanto às mudanças de escopo durante o desenvolvimento do projeto. Desse modo, a cada iteração, os requisitos podem ser refinados.

3.2 PLANO DE ATIVIDADES

O plano de atividades mostra o fluxo de trabalho de acordo com a metodologia de desenvolvimento adotada pela equipe. Esse plano pode ser visto através do diagrama *Work Breakdown Structure* (WBS) e do diagrama de Gantt, que serão apresentados nas sessões seguintes.

3.2.1 Diagrama WBS

O diagrama WBS auxilia na divisão do trabalho, formando uma estrutura hierárquica que fornece uma visão geral das etapas deste trabalho.

O projeto foi dividido em quatro fases, de acordo com o RUP e possui uma quinta fase que corresponde ao gerenciamento de projeto. Essa divisão pode ser observada na FIGURA 14, que corresponde a WBS do projeto.

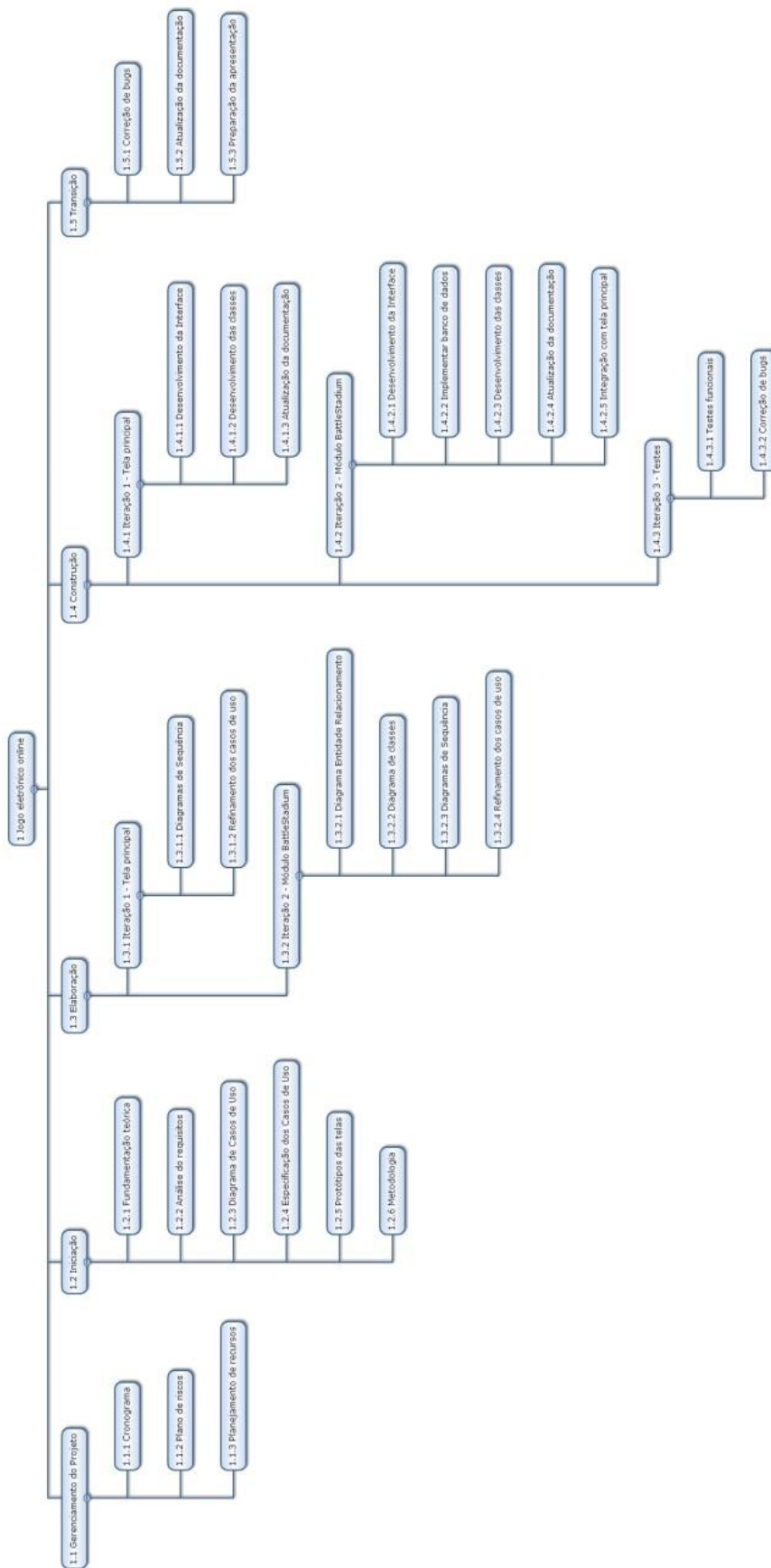



FIGURA 14 – *Work Breakdown Structure*

3.2.2 Diagrama de Gantt

O diagrama de Gantt mostra as atividades de um projeto, quando cada uma deve acontecer e quem irá executá-las. O diagrama de Gantt foi elaborado de acordo com a WBS, e auxiliou no gerenciamento das tarefas e dos recursos do projeto. No diagrama de Gantt, FIGURA 15, foram definidas as atividades no projeto, o prazo de entrega de cada uma e as pessoas responsáveis por elas.

		Nome	Duração	Início	Término	Predecessoras	Nome do Recurso
1		**Jogo eletrônico online**	67 dias?	**05/09/13 08:00**	**06/12/13 17:00**		
2		**Gerenciamento do Projeto**	4 dias	**05/09/13 08:00**	**10/09/13 17:00**		
3		Cronograma	3 dias	05/09/13 08:00	09/09/13 17:00		Carla
4		Plano de riscos	3 dias	05/09/13 08:00	09/09/13 17:00		Lucas B
5		Planejamento de recursos	4 dias	05/09/13 08:00	10/09/13 17:00		Daniel;Lucas S
6		**Iniciação**	51 dias?	**05/09/13 08:00**	**14/11/13 17:00**		
7		Fundamentação teórica	18 dias	12/09/13 08:00	07/10/13 17:00		Alisson;Carla;Daniel;Lucas B;Lucas S
8		Análise do requisitos	2 dias	05/09/13 08:00	06/09/13 17:00		Alisson;Carla;Daniel;Lucas B;Lucas S
9		Diagrama de Casos de Uso	2 dias?	09/09/13 08:00	10/09/13 17:00	8	Carla
10		Especificação dos Casos de Uso	5 dias	10/09/13 08:00	13/09/13 17:00	9	Carla;Daniel
11		Protótipos das telas	4 dias	23/09/13 08:00	26/09/13 17:00	9	Alisson;Lucas B;Lucas S
12		Metodologia	27 dias	09/10/13 08:00	14/11/13 17:00		Carla;Daniel
13		**Elaboração**	14 dias?	**27/09/13 08:00**	**16/10/13 17:00**		
14		**Iteração 1 - Tela principal**	5 dias?	**27/09/13 08:00**	**03/10/13 17:00**		
15		Diagramas de Sequência	3 dias?	27/09/13 08:00	01/10/13 17:00	11	Carla;Daniel
16		Refinamento dos casos de uso	4 dias?	01/10/13 08:00	03/10/13 17:00	15	Carla;Daniel
17		**Iteração 2 - Módulo BattleStadium**	11 dias?	**02/10/13 08:00**	**16/10/13 17:00**		
18		Diagrama entidade relacionamento	3 dias?	02/10/13 08:00	04/10/13 17:00		Lucas B;Lucas S
19		Diagrama de classes	3 dias?	05/10/13 08:00	09/10/13 17:00	9	Carla;Daniel
20		Diagramas de Sequência	5 dias?	09/10/13 08:00	15/10/13 17:00		Carla;Daniel
21		Refinamento dos casos de uso	3 dias?	12/10/13 08:00	16/10/13 17:00		Carla;Daniel
22		**Construção**	42 dias?	**01/10/13 08:00**	**27/11/13 17:00**		
23		**Iteração 1 - Tela principal**	15 dias?	**01/10/13 08:00**	**21/10/13 17:00**		
24		Desenvolvimento da Interface	7 dias?	01/10/13 08:00	09/10/13 17:00		Alisson;Lucas B;Lucas S
25		Desenvolvimento das classes	5 dias?	10/10/13 08:00	16/10/13 17:00	19	Alisson;Lucas B;Lucas S
26		Atualização da documentação	3 dias?	17/10/13 08:00	21/10/13 17:00		Carla;Daniel
27		**Iteração 2 - Módulo BattleStadium**	29 dias?	**14/10/13 07:00**	**21/11/13 17:00**		
28		Desenvolvimento da Interface	8 dias?	17/10/13 08:00	28/10/13 17:00		Alisson;Lucas B;Lucas S
29		Implementar banco de dados	6 dias?	25/10/13 08:00	01/11/13 17:00		Alisson;Lucas B;Lucas S
30		Desenvolvimento das classes	14 dias	28/10/13 08:00	14/11/13 17:00		Alisson;Lucas B;Lucas S
31		Atualização da documentação	2 dias?	14/10/13 07:00	15/10/13 17:00		Carla;Daniel
32		Integração com tela principal	5 dias?	15/11/13 08:00	21/11/13 17:00	30	Alisson;Lucas B;Lucas S
33		**Iteração 3 - Testes**	4 dias	**22/11/13 08:00**	**27/11/13 17:00**		
34		Testes funcionais	2 dias	22/11/13 08:00	25/11/13 17:00	32	Alisson;Lucas B;Lucas S
35		Correção de bugs	3 dias	25/11/13 08:00	27/11/13 17:00		Alisson;Lucas B;Lucas S
36		**Transição**	7 dias	**28/11/13 08:00**	**06/12/13 17:00**		
37		Correção de bugs	4 dias	28/11/13 08:00	03/12/13 17:00		Alisson;Lucas B;Lucas S
38		Atualização da documentação	4 dias	28/11/13 08:00	03/12/13 17:00		Carla;Daniel
39		Preparação da apresentação	4 dias	03/12/13 08:00	06/12/13 17:00		Alisson;Carla;Daniel;Lucas B;Lucas S

FIGURA 15 – Diagrama de Gantt - Divisão dos Recursos

A FIGURA 16 representa uma linha do tempo que possui as tarefas da FIGURA 15 (anterior).

FIGURA 16 – Diagrama de Gantt – Linha do Tempo

3.3 PLANO DE RISCOS

No plano de riscos foram definidos os impactos e a probabilidade de determinados eventos acontecerem, bem como as possíveis ações que poderiam ser tomadas. O plano de riscos do projeto pode ser observado na TABELA 2.

Nº	Risco	Consequência	Ação	Probabilidade	Impacto	Classificação
1	Mudança de requisitos do jogo.	Necessário mudar o direcionamento do projeto.	Mudança e estruturação do projeto.	Alto	Alto	7
2	Falta de Tempo para o desenvolvimento do projeto.	Atraso na entrega ou impossibilidade de entregar.	Monitoramento do andamento.	Alto	Alto	7
3	Perda de Arquivos	Impossibilidade de realizar o projeto	Realizar uma rotina de backup dos arquivos importantes	Moderado	Alto	6
4	Infraestrutura insuficiente para a execução do projeto	Falta de equipamentos e ambiente para o desenvolvimento e teste do jogo	Virtualização do ambiente	Moderado	Moderado	4
5	Desistência de um membro da equipe	Atraso no cronograma do projeto, necessidade de revisar a divisão do trabalho.	Refazer a divisão de tarefas, aumentando a tarefa de cada membro.	Moderado	Moderado	4
6	Conhecimento limitado na área de desenvolvimento de jogos	Atraso no projeto e dificuldade na manipulação no desenvolvimento do jogo.	Treinamento para a equipe, com pesquisas e buscas na internet e livros.	Moderado	Baixo	4

TABELA 2 – Plano de Riscos

3.4 MATERIAIS

O objetivo deste capítulo é descrever as ferramentas utilizadas no desenvolvimento do projeto.

3.4.1 Ambiente de Desenvolvimento

Para o desenvolvimento e testes do jogo, foram utilizados dois sistemas operacionais: Linux na distribuição Ubuntu 11, que serviu como base de desenvolvimento e também como testes. E o outro cliente foi o Windows 7, versão Ultimate, na plataforma de 64 bits, como forma de garantir a portabilidade do jogo entre os sistemas operacionais. O servidor utilizado neste projeto foi um Linux na distribuição Ubuntu Server 13.04. Há no contexto a utilização do MySQL, do GitHub, e da Internet como meio de ligação entre os sistemas, e a função do jogo *online* por intermédio de dois clientes ligados via LAN. A FIGURA 17 demonstra graficamente estas conexões:

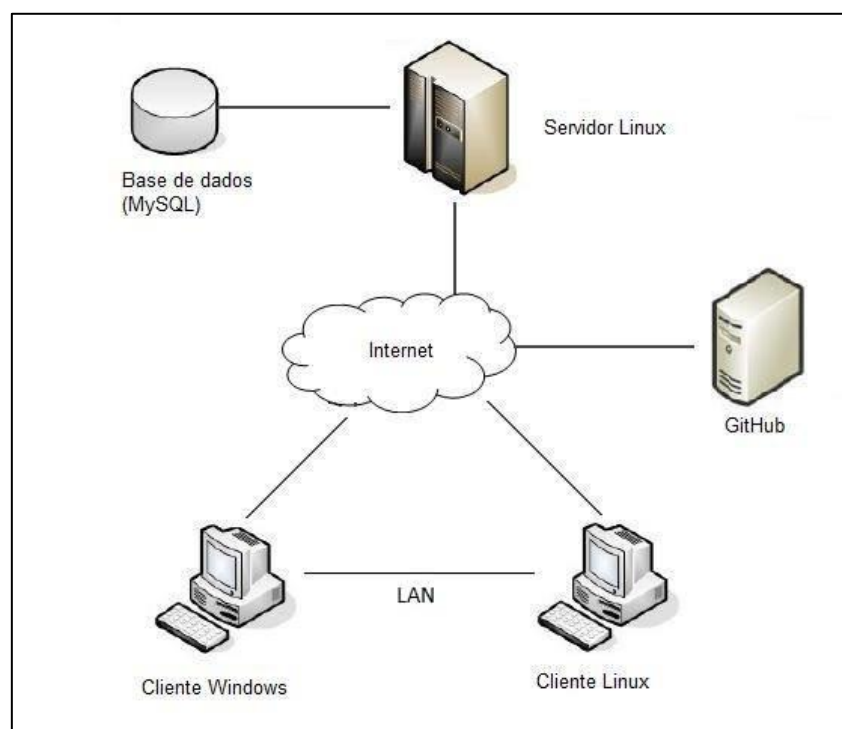



FIGURA 17 – Ambiente de Desenvolvimento.

3.4.2 Linguagem de Programação

A linguagem de programação utilizada neste trabalho foi Java. Essa linguagem é baseada na orientação a objetos e pode ser usada em diferentes plataformas. Segundo a Oracle, empresa detentora dos direito desta linguagem atualmente, Java é uma linguagem de programação e uma plataforma de computação lançada pela primeira vez em 1995 pela Sun Microsystems. É a tecnologia que capacita muitos programas da mais alta qualidade, como utilitários, jogos e aplicativos corporativos, entre muitos outros, por exemplo. O Java é executado em mais de 850 milhões de computadores pessoais e em bilhões de dispositivos em todo o mundo, inclusive telefones celulares e dispositivos de televisão. (JAVA, 2013).

A linguagem Java foi escolhida devido aos membros da equipe possuírem maior conhecimento nesta linguagem. Outra vantagem da utilização desta linguagem é o fato de possuir um grande material sobre ela.

3.4.3 Sistema Gerenciador de Banco de Dados

O sistema gerenciador de banco de dados (SGBD) escolhido para este projeto foi o MySQL, que é um software de código aberto. Uma das vantagens desse SGBD é o seu bom desempenho e estabilidade. O MySQL possui uma grande flexibilidade entre plataformas, visto que o projeto foi desenvolvido em dois sistemas operacionais diferentes. O sistema é segundo a própria companhia: “Uma das mais utilizadas ferramentas de código aberto para gerenciamento de banco de dados relacionais, sendo desenvolvido, distribuído e suportado pela Oracle.” (MySQL, 2013, tradução nossa).

3.4.4 Ambiente Integrado de Desenvolvimento

A ferramenta Eclipse foi utilizada neste projeto como IDE (*Integrated Development Environment*). O Eclipse possui licença gratuita e suporta diversas linguagens como Java, C/C++.

A IDE Eclipse foi a escolhida devido aos membros da equipe já possuírem experiência com esta ferramenta e por ela ser multiplataforma.

3.4.5 Ferramenta para criação do diagrama WBS

A ferramenta utilizada para elaborar o diagrama WBS do projeto foi a WBS *Tool*. Essa ferramenta é um *software* gratuito que disponibiliza as funcionalidades necessárias para criar diagramas WBS e organogramas.

3.4.6 Ferramenta para criação do Gantt

A ferramenta utilizada para elaborar o Gantt do projeto foi a *OpenProject*, um *software* livre e de código aberto para auxiliar no gerenciamento de projetos.

3.4.7 Ferramenta para Modelagem UML

Para auxiliar na modelagem do jogo foi escolhido o *software* AstahCommunity, que possui suporte à linguagem UML 2.0. Esse *software* é uma versão gratuita disponibilizada pelo grupo Astah. Como a versão utilizada é gratuita, a ferramenta possui algumas limitações de recursos, porém atendeu às necessidades.

3.5 FERRAMENTA DE CONTROLE DE VERSÃO

O Git é uma ferramenta utilizada principalmente para gerenciar versões de softwares desenvolvidos por um ou mais desenvolvedores. Todos os integrantes de um projeto podem enviar alterações, mas somente o dono do projeto pode incluir essas alterações que são sempre registradas em um histórico. Todos os arquivos e históricos do projeto ficam armazenados em um local do computador, chamado de repositório.

Existem sites que servem como repositórios *online* (os arquivos ficam hospedados na Internet) como, por exemplo, o GitHub, que utiliza o controle de versionamento do Git. O GitHub foi utilizado neste projeto como repositório *online* dos códigos, facilitando o acesso das últimas versões do projeto à todos os membros da equipe.

3.6 FERRAMENTA PARA CRIAÇÃO DE SPRITES

Para criar os *sprites* do jogo, foi utilizado o *Allegro Sprite Editor*, um software de código aberto que permitiu que fizéssemos as animações dos jogos e dos personagens, por possuir uma biblioteca rica e que permitiu-nos uma maior liberdade de criação.

3.7 BIBLIOTECAS

Na programação em linguagens de computadores existe a possibilidade do uso de um conjunto de funções pré-escritas, onde vários problemas já estão mapeados e resolvidos, são neles em que os programadores recorrem para obter suporte para que não precisem criar algo já existente, este conjunto de funções facilitadoras é denominado de biblioteca. Isso pode ser um fator de extrema importância quando se tratar de programas de grande porte onde os códigos são muito extensos ou complexos.

Através de pesquisas revelaram-se as melhores opções, o uso das bibliotecas Slick para o desenvolvimento de jogos de segunda dimensão na linguagem Java, e a X-Stream para a manipulação do formato XML.

3.7.1 Slick-Util

A Slick-Util é uma biblioteca de programação que possibilita o carregamento de várias imagens, sons e formatos de fontes para o uso em determinadas áreas. Ela é uma alternativa de fácil uso e de tamanho leve, a qual foi selecionada por ser um caminho simples para obtenção do que se esperava em termos de tratamento de imagens no jogo.

3.7.2 X-Stream

A X-Stream é uma biblioteca de programação que foi criada para facilitar a manipulação de XML em Java, através da serialização e deserialização de objetos, que é o nome dado ao processo de transformação de objetos em XML e seu caminho inverso, que é o XML em objetos, de qualquer maneira é um recurso que possibilita uma fácil interpretação e uso.

4 DESENVOLVIMENTO DO PROJETO

A origem do projeto surgiu da ideia da equipe de implementar um jogo para ser disputado numa rede de acesso local e em modo *offline,* nos moldes de um jogo chamado Bomberman, um dos títulos de maior sucesso da décadas de 80 e 90, pelo fato da descontinuidade de tal jogo e a falta da implementação de novos mapas e um modo multijogadores.

De acordo com as características principais do jogo, foram levantados os requisitos e, conjuntamente, o desenvolvimento do projeto do jogo em si, contendo uma elaboração de um plano de trabalho. Com isso, fora decidido pela equipe a realização de uma pesquisa de campo para ter uma base dos sistemas já desenvolvidos e o que ainda carecia em tais sistemas. Após a pesquisa, vimos a necessidade de um jogo mais simples, resgatando um pouco da jogabilidade e gráficos antigos, mas em um nova plataforma, com mais recursos.

Para a implementação do jogo fora necessário a pesquisa de ferramentas gráficas. Dessa pesquisa, devido a todas as funcionalidades e fácil manuseio, utilizamos o software livre Allegro Sprite Editor, para edição dos gráficos de fundo, animação dos personagens, itens, cenário, como também utilizando uma base pública disponível no site “The Spriters Resource”. Também, para o processamento e carregamento dos gráficos, utilizamos a biblioteca Slick-Util, que se trata de uma pequena biblioteca que habilita ao desenvolvedor carregar imagens, sons e fontes. Assim como para a transmissão desses dados via rede, fora utilizado a biblioteca X-Stream, para serializar os objetos e enviá-los via rede.

Para o desenvolvimento, redistribuição e teste do projeto, se viu necessário utilizar um servidor que pudesse armazenar todo o sistema, mantendo o versionamento dos arquivos e facilitando a redistribuição do código-fonte de maneira rápida e segura. Para tal feito a equipe decidiu em utilizar o Github, um servidor *online* para compartilhamento de código com versionamento.

Dentro os requisitos do sistema estão:

- **Login de usuário:** Uma seção na página principal que possibilita ao usuário o cadastro e login para utilizá-lo em partidas em rede local;
- **Tela de configuração do jogo offline e em rede local:** Tela de configuração das regras da partida;
- **Módulo de jogo Battlestadium offline e em rede local:** Implementação do jogo segundo as regras definidas na tela de configuração da partida;
- **Tela com pontuação das partidas feitas em redes locais:** Tela contendo o *ranking* das pontuações de todos os usuários segundo as partidas efetuadas via rede local;
- **Tela de conquistas:** Tela com uma lista de conquistas a serem desbloqueadas. À medida que o usuário joga nas partidas *online*, desbloqueia troféus de acordo com as ações realizadas na partida, por exemplo, ganhar o primeiro troféu na partida (que equivale a uma rodada);
- **Opções do Sistema:** Tela de configurações do sistema contendo o mapeamento das teclas, volume, tamanho da tela, exibição em tela cheia.

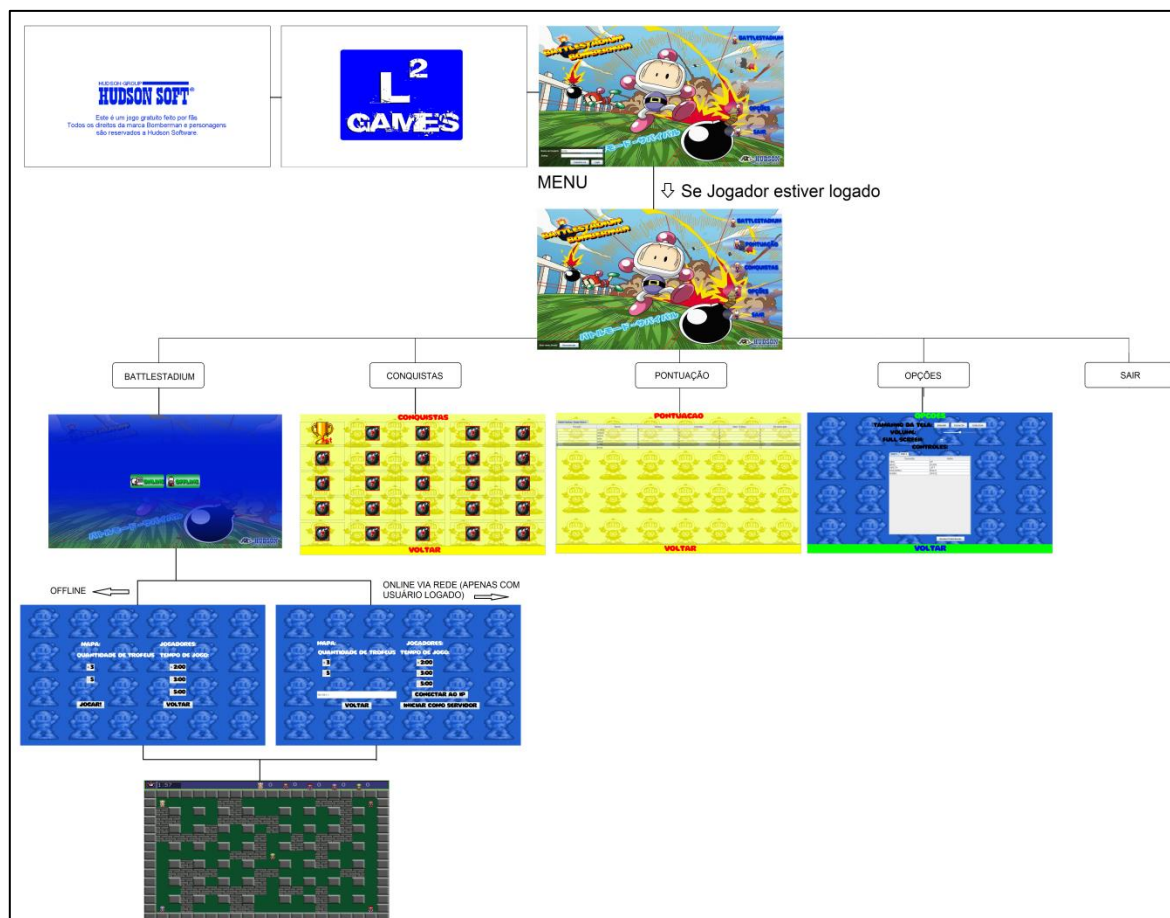



FIGURA 18 – Diagrama de telas do jogo

De acordo com a FIGURA 18, que mostra o diagrama de telas do jogo, e com o ambiente e as tarefas definidas e compartilhadas, começou a definição da modelagem de todo o projeto em conjunto com toda a documentação necessária para ampliar a visão do projeto e facilitar a implementação do jogo como um todo. Documentação essa que pode ser encontrada nos apêndices deste trabalho, de acordo com a TABELA 3, contendo partes importantes para a estruturação do da modelagem do sistema como diagramas de casos de uso, mostrando as diversas funcionalidades do sistema.

APÊNDICE	DESCRIÇÂO
A	DIAGRAMA DE CASO DE USO
B	ESPECIFICAÇÃO DOS CASOS DE USO
C	DIAGRAMA ENTIDADE RELACIONAMENTO
D	DIAGRAMA DE CLASSES
E	DIAGRAMA DE SEQUÊNCIA

TABELA 3 - Apêndices

5 APRESENTAÇÃO DO JOGO BATTLESTADIUM

Neste capítulo será apresentada uma análise do jogo desenvolvido, os protótipos e funcionalidades de cada tela do jogo e um manual de instalação do sistema.

5.1 VISÃO GERAL

Foi desenvolvido um jogo multiplataforma e multijogador criado na linguagem Java, podendo ser disputado pela rede local (LAN) ou em modo de jogo *offline*.

O gênero deste jogo é estratégico com labirintos, que consiste em séries de rodadas aonde só é possível à conclusão da fase através de posicionar bombas em lugares estratégicos para derrotar inimigos, outros jogadores e desobstruir o caminho.

O jogo requer uma conexão com a Internet e *login* do usuário para que o módulo BattleStadium possa ser disputado de forma *online*.

5.1.1 Cadastro de Usuário

O cadastro de usuário é uma tela na qual o jogador tem a possibilidade de criar uma conta para que possa jogar partidas multijogadores. Os benefícios dessa conta, além de poder se conectar com outros jogadores e jogar em diferentes equipamentos, é que ele pode receber notícias sobre atualizações no e-mail fornecido neste cadastro, bem como visualizar sua pontuação das partidas em que disputou.

5.1.2 Módulo BattleStadium

O módulo BattleStadium é um estilo de jogo onde o objetivo do jogador é somar um determinado número de vitórias pré-determinados dentro de um intervalo ambos definidos em um menu de escolha. O modo de jogo contém os seguintes parâmetros para serem configurados:

- O módulo terá a capacidade de até cinco jogadores simultâneos no modo *offline*, e dois jogadores no modo *online*.
- As batalhas poderão contemplar o mínimo de três partidas, com no máximo de cinco rodadas disputadas entre os jogadores;
- A duração de cada partida é de no máximo cinco minutos, e no mínimo de dois minutos. O módulo é disponível em rede local e *offline*.

5.2 FUNCIONALIDADES

Nesta seção serão apresentadas as funcionalidades do jogo BattleStadium, através dos protótipos das telas. O jogo possui uma tela inicial que permite que o jogador acesse os módulos do jogo, cadastre-se e efetue o *login*, tela de opções do sistema, tela que mostra as conquistas do jogador, a tela do módulo BattleStadium.

5.2.1 Tela Inicial

Na tela inicial do jogo, que pode ser vista na FIGURA 19, é possível fazer um cadastro, através da opção “cadastre-se”, e o efetuar *login*, informando o nome de usuário e senha, que devem estar previamente cadastrados. Antes de efetuar o *login* só é possível acessar o menu BattleStadium (no modo *offline*) e alterar as configurações do sistema no menu “Opções”. Após realizar o *login*, é possível acessar a "Pontuação" e "Conquistas", além do modo *online*, como pode ser observado na FIGURA 20.

FIGURA 19 – Tela Inicial

FIGURA 20 – Tela Inicial 2

5.2.2 Tela de Configuração do Sistema

O jogador terá a possibilidade, de acordo com as suas preferências pessoais, de alterar algumas opções do jogo. Ao entrar no jogo, existirá uma

alternativa chamada “Opções”, como exibido na FIGURA 21, na qual contemplará as seguintes preferências:

- **Volume:** O jogador terá a possibilidade de regular o volume do jogo, assim determinando uma intensidade sonora ideal.
- **Vídeo:** Serão exibidas três opções de resolução (800x600, 1024x728, 1280x1024), as quais o jogador poderá selecionar a mais apropriada para o seu equipamento.
- **Controle:** Refere-se à configuração de controles do jogo, haverá descrito em abas *“Pads”*, que são os dispositivos de jogo. Então haverá a descrição à esquerda do que faz o personagem (cima, baixo, esquerda, direita, bomba) e à direita quais teclas ou botões serão os responsáveis pelas ações.

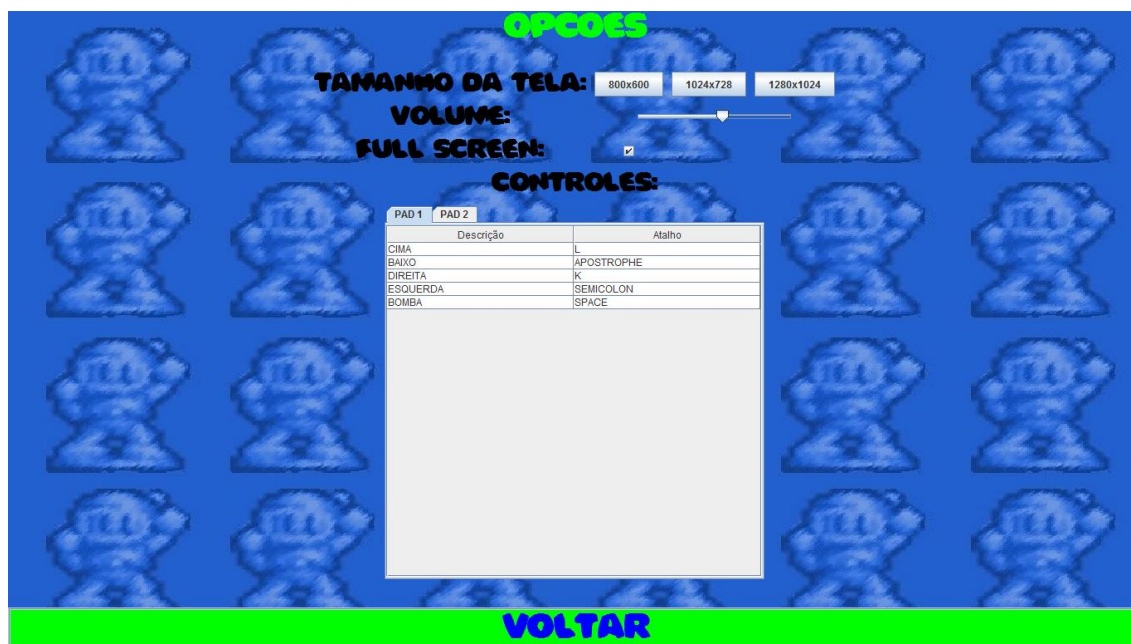



FIGURA 21 – Opções do jogo

5.2.3 Conquistas (Achievements)

As conquistas dos jogadores são alguns objetivos dentro do jogo alcançados pelos usuários, como forma de motivação. Há uma parte no menu principal chamada de “Conquistas” que são as conquistas do usuário dentro do

jogo, como pode ser exemplificado na FIGURA 22, então um *hall* será apresentado, como se fosse uma parede de prêmios, com troféus e outros ainda que não foram alcançados, e ao selecionar um troféu, como na FIGURA 23, pode ser exibido o objetivo alcançado com aquele prêmio.

FIGURA 22 – Conquistas

FIGURA 23 – Conquistas 2

5.2.4 Pontuação

A pontuação do jogo, com a entrada localizada na tela principal. O BattleStadium Single Match refere-se ao modo de jogo individual, aonde o estilo "todos-contra-todos" é utilizado. Como exibido na FIGURA 24, existe uma tabela na qual são exibidos os seguintes itens:

- **Posição:** O item posição é referente a posição do jogador em detrimento aos demais jogadores, isto é calculado em relação à quantidade superior do "Bombersaldo", quanto maior o número deste em relação aos demais, superior será a sua posição na tabela de pontuação.
- **Nome:** É exibido o nome do jogador.
- **Vitórias:** São as partidas ganhas em cada módulo, de forma individual a cada partida finalizada por tempo ou por eliminação de todos os outros participantes do jogo.
- **Derrotas:** São as partidas onde algum outro jogador venceu, e portanto, calcula-se uma derrota na pontuação do jogador.
- **Qtde. Troféus:** A quantidade de troféus significa que um jogador conseguiu, por padrão do jogo, venceu três partidas por primeiro. Assim o jogador conseguirá um troféu por ter conquistado as vitórias contra o adversário, seja em rede local ou *offline*, em qualquer módulo do jogo.
- **Bombersaldo:** O bombersaldo segue uma regra para definir a pontuação do jogador. Este item é calculado da seguinte maneira: ((número de vitórias * 3) + (número de derrotas * (-1) + (qtde de troféus * 2)). Como exemplo o jogador "lucas", no modo BattleStadium Single Match, na FIGURA 23, têm 10 vitórias, nenhuma derrota e 15 troféus, o bombersaldo será ((10 * 3) + (15 * 2)), portanto (30+30), e seu bombersaldo será de 60 pontos.

FIGURA 24 - Pontuação do módulo BattleStadium Single Match

5.2.5 Tela Online e Offline

No modo de batalha BattleStadium, o usuário poderá escolher se deseja jogar no modo *online* (rede local) ou *offline*, como exemplificado na FIGURA 25. No modo *online* o jogador terá a oportunidade de jogar com outros jogadores que estejam na mesma rede, mas em outras máquinas. No modo *offline* o jogador poderá realizar um jogo com outros jogadores no mesmo equipamento, com controles de jogos configurados previamente para cada jogador.

FIGURA 25 - Tela de escolha *online* e *offline*

5.2.6 Tela de Configuração Online

Após o jogador ter escolhido a partida no modo *online*, o jogo automaticamente o direcionará para uma tela de configuração para a partida *online*, como demonstrado na FIGURA 26. Esta tela exibirá as seguintes opções para o jogador:

- **Quantidade de troféus:** O jogador poderá selecionar quantas partidas o jogo poderá ter. São duas opções: Três ou cinco. Quem conquistar primeiro a quantidade de disputas configurada, ganha o jogo.
- **Tempo de jogo:** Nesta opção será possível configurar o tempo que cada rodada terá, e terá três opções: dois, três ou cinco minutos.
- **Conectar ao IP:** Aqui o jogador poderá digitar o IP do parceiro que está configurado como servidor. Após a digitação, o jogo entrará automaticamente na partida.
- **Iniciar como servidor:** Quando o jogador deseja ser o servidor da partida, então selecionará esta opção, e o jogo aguardará que mais clientes se conectem a ele.

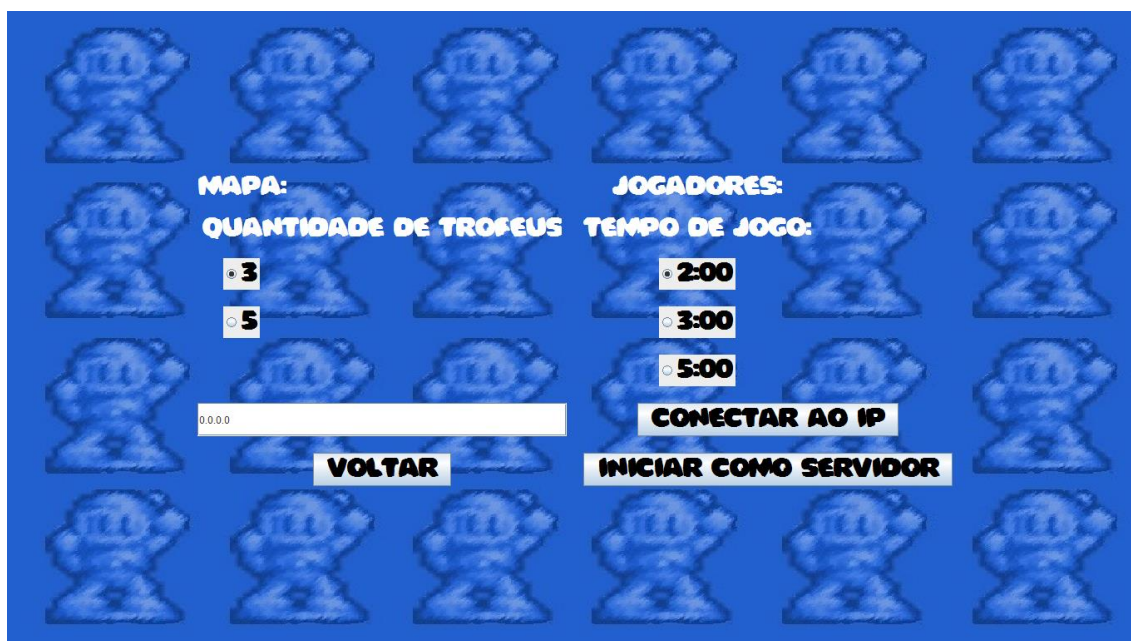



FIGURA 26 – Tela de configuração online

5.2.7 Tela de Configuração Offline

Após a escolha pelo jogador ter sido *offline*, ou seja, que ele possa jogar no mesmo equipamento com outros usuários, o jogo de forma automática irá direcioná-lo para a tela de configuração *offline*. A FIGURA 27 demonstra a tela na qual possuirá duas configurações, a da quantidade de troféus e o tempo de jogo, assemelhando-se às configurações *online* em funções.

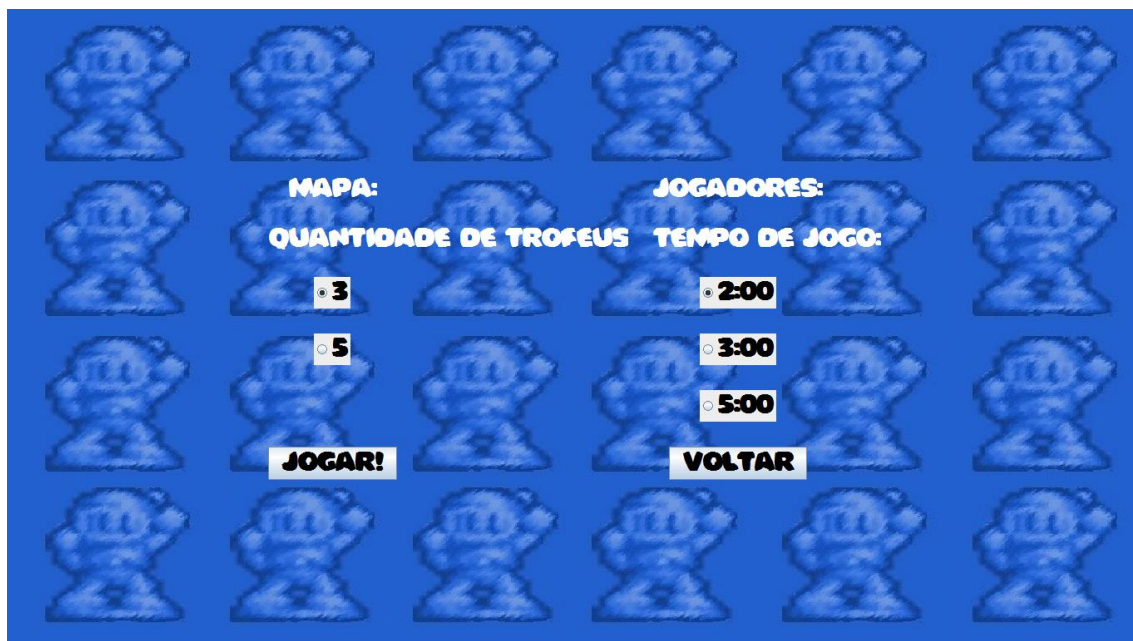



FIGURA 27 – Tela de configuração *offline*

5.2.8 Tela do Jogo

O jogo possuirá um padrão para a posição inicial dos jogadores, como exemplificado na FIGURA 28. Em uma partida com cinco jogadores, quatro ficarão em posições diagonais superiores e inferiores do jogo, e um dos jogadores ficará no centro do mapa, para uma melhor distribuição geral.

O jogador poderá se mover para cima, baixo e para direita e esquerda, de acordo com a estratégia definida individualmente ou por grupo. Dentre os movimentos, o jogador deverá plantar uma bomba, que terá um símbolo semelhante à FIGURA 29, o qual poderá eliminar um jogador ou objeto que estiver ao seu alcance. Nem todos os objetos poderão ser destruídos, apenas os objetos que tiverem distinção de cor ou diferença de tonalidade com o mapa em si. Desta forma o jogador abrirá caminho no mapa para alcançar outros oponentes e conquistar itens extras ao explodir os itens, como o poder de: deixar duas ou mais bombas pelo caminho, aumentar o poder da bomba (que explodirá em formato de cruz, formando chamas que eliminarão ao alcance, como exibido na FIGURA 30), um ícone com formato de patins que será para

aumentar a velocidade de movimentação do jogador, a FIGURA 31 exibe graficamente estes itens em modo *sprite*.

FIGURA 28 - Tela do jogo

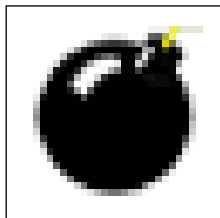



FIGURA 29 - Bomba no jogo

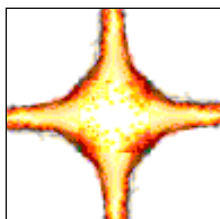



FIGURA 30 - Chamas após a explosão da bomba

FIGURA 31 - Bônus durante o jogo (de cima para baixo - bomba adicional, maior explosão e velocidade de movimentação).

5.2.9 Tela da Vitória

Após as vitórias do jogador, ele receberá um ponto na quantidade de troféus na pontuação, e também será exibida uma tela confirmando que o jogador foi o vencedor da batalha. Cada bomba, na FIGURA 32, simboliza uma vitória e um texto aparecerá descrevendo a vitória do jogador e também o personagem que foi utilizado na batalha.

FIGURA 32 - Tela da vitória

5.3.1 Instalação

O jogo BattleStadium foi criado de forma que seja necessária uma conexão com a Internet para o funcionamento do modo *online,* requisitando assim um servidor de aplicação, um banco de dados para a contabilização das pontuações e cadastros, e o jogo em si instalado. Para o modo *offline,* a qual não é necessário o *login*, tendo apenas que possuir o jogo instalado, o usuário poderá configurar localmente os seus controles para a disputa.

5.3.1.1 Instalação do servidor de aplicação

O servidor de aplicação, para a execução de cadastro e gerenciamento de usuários, está hospedado no endereço de IP “200.236.3.203”. Este endereço está predefinido no projeto, dentro do arquivo “pres.xml” localizado na raiz, pelo parâmetro “ip_tomcat”, que poderá ser alterado caso seja necessário.

Ou como forma alternativa, o *deploy* do arquivo “bbServer.war”, em um servidor Tomcat, já instalado e configurado, com o usuário com direitos administrativos.

Para a instalação de um servidor de aplicação local, os seguintes procedimentos poderão ser realizados no programa Eclipse Kepler:

- Dentro do programa Eclipse, entre em “*File*” e após *“Import”*.
- Abra a aba *“General”* e escolha *“Existing Project into Workspace”*, e clique em *“next”.*
- Escolha o diretório onde está localizada a pasta do projeto. Note que a IDE do Eclipse reconhecerá o projeto, e clique em *“Finish”,* e o projeto aparecerá na aba *“Project Explorer”.*

Para a configuração do servidor de aplicação local, dentro do Eclipse, os seguintes procedimentos poderão ser tomados:

- Na aba *"Servers"*, clique com o botão direito do mouse e escolha *"New"* e *"Server"*.
- Escolha o servidor, que no caso deste projeto será o Tomcat na sua versão 7, e clique em *"next".*
- Escolha o diretório onde instalou o *server,* clique em *"next".*
- Após isto, aparecerá uma tela para a escolha de quais projetos deverão ser adicionados ao servidor de aplicação. Escolha o projeto que foi importado e clique em *"Add".*

5.3.1.2 Script do banco de dados

O banco de dados MySQL já está hospedado no servidor de aplicação citado anteriormente. Para a instalação do banco de dados em um servidor de aplicação diferente, serão necessários os seguintes procedimentos:

- Pode ser feita a extração do banco de dados diretamente do servidor já criado (200.236.3.203) pelo comando "`mysqldump –u dbusr –p bb > exportdatabase.sql`", que exportará o arquivo de *dump* do banco de dados para o diretório atual do comando.
- Caso o usuário já possua o arquivo de *dump* do banco de dados, ou tenha feito o comando anterior, será necessário apenas a execução do comando "`mysqldump –u "usuário" -p < arquivodedump.sql`" para a inserção da estrutura do banco de dados no servidor de aplicação desejado.

5.3.1.3 Instalação do projeto no cliente

Após os procedimentos anteriores de instalação, o usuário poderá instalar o jogo BattleStadium no equipamento. Para efetuar a instalação, os seguintes procedimentos poderão ser tomados:

- Efetuar o download do Eclipse Kepler no site, na sua versão mais atual, e instalar no equipamento.
- Instalação do *plugin* Egit, disponível no site do Eclipse, para que possa ser feito o download do projeto.
- Após a instalação do Eclipse, e seu plugin Egit, proceder com a importação do projeto, no Eclipse: Clicar em "*file*", "*Import*", "Git", "*Projects from* Git" e clique em "*next*".
- Na próxima janela, selecionar o "Clone URI" e clique em "*next*" e digitar o seguinte URI: < https://github.com/tcc5/bb_project.git >, e clique em "*next*".
- Na próxima tela, deixe marcado a opção *"master",* e clique em "*next*".
- Na próxima janela escolha o local onde será armazenado o repositório do GitHub e clique em *"next".*
- A tela seguinte selecione a opção *"Import existing project".*
- Após o *download*, selecione o projeto "BB" para a importação e clique em *"finish".*

5.3.3 Acesso ao Sistema

Para acesso ao cliente do jogo, no menu principal há a opção de efetuar um *login*, necessário para participar de partidas em rede local e comparar a pontuação com outros jogadores.

Dentro do jogo há uma sessão para cadastro, FIGURA 33, de novos jogadores, com um formulário que solicita as informações de nome de usuário, e-mail, senha, confirmação da senha, sendo todos os campos validados para que seja feita a devida inserção na base de dados. Para acesso como usuário, pode se utilizar a seguinte conta genérica, sendo o *login* "bomb", com a senha "bomb", sem as aspas.

FIGURA 33 - Tela de cadastro de usuário

6 CONSIDERAÇÕES FINAIS

O objetivo deste projeto de desenvolver um jogo de estratégia e ação, que fosse multiplataforma e multijogador, que pudesse atender às necessidades dos fãs do antigo jogo Bomberman, foi atendido. Entre as funcionalidades do jogo com as propostas atendidas estão o cadastro de usuários, visualização de pontos alcançados, implementação do jogo *offline* e *online*, e o desenvolvimento de interfaces de configurações do jogo e gerenciamento de usuários.

As técnicas de modelagem de sistemas aplicadas neste projeto auxiliaram na organização e divisão das tarefas pela equipe, assistindo aos membros uma visão dos objetivos propostos antes da programação do jogo em

Uma proposta de trabalho futuro seria o aperfeiçoamento da interface gráfica do jogo, assim como a criação de outros mapas maiores e mais desafiadores aos jogadores, proporcionando um maior nível de satisfação do jogador em relação ao jogo atual. A implementação de um servidor de aplicação daria a possibilidade de vários usuários jogarem em diferentes locais, por intermédio da Internet, possibilitando assim partidas em longas distâncias entre jogadores. Também seria interessante a criação de um site para o gerenciamento do jogo, com a criação de um *Web Service* próprio, que possuiria a função de cadastro de usuários, visualização da pontuação dos jogadores (*ranking*), desafios *online* de jogos por bate-papo, geração de relatórios dos jogos e jogadores pelo administrador do sistema, ocasionando um maior uso do jogo, que poderia ter desafios entre times de jogadores, e competições oficiais do jogo.

REFERÊNCIAS

ARANHA, G. **O processo de consolidação dos jogos eletrônicos como instrumento de comunicação e de construção de conhecimento.** 2004. Ciências & Cognição; Ano 01, Vol 03.

PEDERSEN, ROGER E. **Game design foundations**. 2002. Wordware Publishing, Inc.

PERUCIA E OUTROS. **Desenvolvimento de Jogos Eletrônicos**. 2005. Novatec.

RUCKER, RUDY V. B. **Software engineering and computer games**. 2002. Addison Wesley.

TANENBAUM, ANDREW S. **Redes de Computadores**. Campus, 4 Edição, 2003.

Abragames - Crescente interesse do público na indústria de jogos eletrônicos Disponível em: <http://www.abragames.org/wp-content/uploads/2013/04/Abragames-Pesquisa_2008.pdf>. Acesso em 09/10/2013.

Biblioteca Slick-Util.
Disponível em: <http://slick.ninjacave.com/slick-util/>. Acesso em: 10/11/2013.

Biblioteca X-Stream.
Disponível em: <http://xstream.codehaus.org/index.html>. Acesso em: 10/11/2013.

IBM – Definição de RUP
Disponível em: <http://www-01.ibm.com/software/rational/rup/>. Acesso em: 15/10/2013.

Instalação do Eclipse Kepler
Disponível em: <http://www.eclipse.org/downloads/>. Acesso em: 23/11/2013.

Instalação do *plugin* Egit
Disponível em: <http://www.eclipse.org/egit/download/>. Acesso em 23/11/2013.

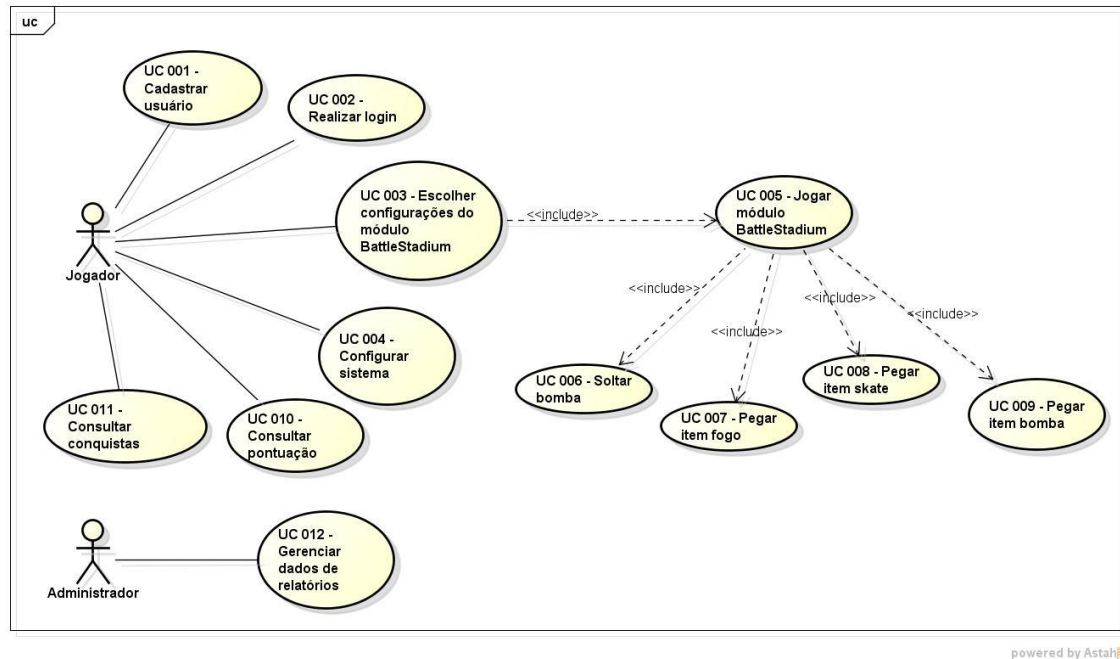
JAVA – Definição e história da linguagem Java.
Disponível em: <http://www.java.com/pt_BR/download/faq/whatis_java.xml>. Acesso em: 17/10/2013.

MySQL – Definição do sistema gerenciador de banco de dados MySQL
Disponível em: <http://dev.mysql.com/doc/refman/4.1/en/what-is-mysql.html>. Acesso em: 18/10/2013.

PEREIRA, GEAN A. **Projeto e Desenvolvimento de Jogos Computacionais.** 2006. Disponível em: <http://www.pergamum.udesc.br/dados-bu/000000/000000000002/0000020E.pdf>. Acesso em: 20/10/2013.

Quake – Código aberto.
Disponível em: <http://www.github.com/id-Software/Quake>. Acesso em: 17/10/2013.

The Spriters Resource – Base pública de dados gráficos.
Disponível em: <http://www.spriters-resource.com/>. Acesso em: 25/11/2013.

APÊNDICE A – DIAGRAMA DE CASOS DE USO

APÊNDICE B – ESPECIFICAÇÃO DOS CASOS DE USO

Caso de Uso 001 – Cadastrar Usuário

Descrição

Este caso de uso descreve o cadastro de usuário no jogo.

Pré – condições

Não há.

Pós – condições

Após o cadastro, o jogador poderá logar no sistema.

Ator(es)

Jogador

Fluxo Principal de Eventos

1. Na tela inicial, o jogador seleciona a opção de cadastro de usuário;
2. O sistema abre o formulário de cadastro;
3. O jogador insere os dados no formulário;
4. Sistema valida os danos inseridos no formulário;
5. Sistema mostra mensagem de confirmação do cadastro;
6. O caso de uso é finalizado.

Fluxo de Exceção

E1 Existe um usuário com o Login digitado

E1.1 Sistema exibe mensagem “Já existe um usuário com este login”.

E1.2 Caso de uso continua no passo 3 do fluxo principal.

E2 Campo “Login” vazio

E2.1 Sistema exibem mensagem “Campo login é obrigatório”;

E2.2 Caso de uso continua no passo 3 do fluxo principal.

E2 Campo “Senha” vazio

E3.1 Sistema exibem mensagem “Campo senha é obrigatório”;

E3.2 Caso de uso continua no passo 3 do fluxo principal.

Regras de Negócio

R1 Preenchimento obrigatório de campos

R1.1 Os campos “Login” e “Senha” são de preenchimento obrigatório.

Caso de Uso 002 - Realizar login

Descrição

Este caso de uso serve para que um usuário possa logar no sistema.

Pré – condições

O jogador deve possuir um cadastro.

Pós – condições

Se o jogador conseguir logar no sistema, poderá acessar as opções: Pontuação e Conquistas;

Ator(es)

Jogador

Fluxo Principal de Eventos

1. Na tela do módulo de jogo escolhido, sistema mostra login e senha;
2. Jogador insere seu login e sua senha referentes ao seu cadastro;
3. Sistema valida os dados inseridos;
4. Caso de uso é finalizado.

Fluxo de Exceção

E1 Login e senha inseridos pelo jogador estão incorretos

E1.1 Sistema exibe mensagem “O login ou a senha estão incorretos”;

E1.2 Caso de uso retorna ao passo 2 do fluxo principal.

Regras de Negócio

R1 Preenchimento obrigatório e válido de campos

R1.1 Os campos “Login” e “Senha” são de preenchimento obrigatório e seu conteúdo deve ser válido.

Caso de Uso 003 – Escolher configurações do módulo BattleStadium

Descrição

Este caso de uso serve para que o jogador possa escolher as configurações disponíveis de cada módulo.

Pré – condições

Para escolher as configurações, o jogador deve ter selecionado um módulo de jogo.

Pós – condições

Não há.

Ator(es)

Jogador

Fluxo Principal de Eventos

1. Na tela do módulo de jogo selecionado, o sistema mostra as opções de jogo: “Criar” e “Entrar sala existente”.
2. O jogador seleciona a opção “Criar”;
3. Sistema abre tela de configurações de acordo com o módulo selecionado;
4. O jogador informa as configurações desejadas;
5. O caso de uso é finalizado.

Fluxo Alternativo

1. Na tela do módulo de jogo selecionado, o jogador seleciona a opção “Entrar sala existente”;
2. Sistema mostra salas existentes;
3. Jogador seleciona a sala desejada;
4. O caso de uso é finalizado.

Fluxo de Exceção

Não há.

Regras de Negócio

R1 Número de partidas

R1.1 O jogador pode escolher jogar no mínimo 3 e no máximo 5 partidas.

R1.2 Número de jogadores

O número de jogadores simultâneos pode ser no mínimo 2 e no máximo 5.

R1.3 Duração da partida

O jogador pode escolher o tempo de partida tendo no mínimo dois minutos e no máximo cinco minutos de duração.

Caso de Uso 004 – Configurar sistema

Descrição

Este caso de uso descreve serve para que o jogador possa escolher as configurações de sistema.

Pré – condições

Não há.

Pós – condições

Não há.

Ator(es)

Jogador

Fluxo Principal de Eventos

1. Na tela inicial, o jogador seleciona a opção “Opções”.
2. O sistema mostra a tela de configurações.
3. O jogador seleciona as configurações desejadas.
4. O caso de uso é finalizado.

Fluxo de Exceção

Não há.

Regras de Negócio

Não há.

Caso de Uso 005 – Jogar Módulo BattleStadium

Descrição

Este caso de uso serve para que o jogador possa jogar o módulo “BattleSadium”.

Pré – condições

O jogador deve ter escolhido as configurações do módulo.

Pós – condições

Não há.

Ator(es)

Jogador

Fluxo Principal de Eventos

1. Sistema inicia a partida;
2. A partida termina quando acabar o tempo de jogo ou o número de partidas;
3. Sistema salva a pontuação do jogador;

4. Caso de uso é finalizado.

Fluxo de Exceção

Não há.

Regras de Negócio

Não há.

Caso de Uso 006 – Soltar bomba

Descrição

Este caso de uso serve para que seja colocada uma bomba quando o jogador apertar certa tecla.

Pré – condições

Não há.

Pós – condições

Não há.

Ator(es)

Jogador

Fluxo Principal de Eventos

1. Na tela do jogo, o jogador aperta certa tecla para soltar uma bomba;
2. Uma bomba é colocada no local onde o personagem que representa o jogador estava quando a tecla foi apertada;
3. Depois de determinado tempo, a bomba “explode”;
4. O caso de uso é finalizado.

Fluxo de Exceção

E1 A bomba, ao explodir atinge uma “parede”

E1.1 A parede atingida é destruída.

E2 A bomba, ao explodir atinge um jogador

E2.1 O jogador atingido morre e sai da partida.

Regras de Negócio

R1 Se o jogador ainda não possuir o “item bomba”, só poderá soltar uma bomba por vez. Para soltar outra bomba, deve esperar a primeira explodir.

Caso de Uso 007 – Pegar item fogo

Descrição

Este caso de uso serve para aumentar o alcance da bomba quando ela explodir.

Pré – condições

No início do jogo, o item fogo deve estar “escondido” em uma “parede”.

Pós – condições

Não há.

Ator(es)

Jogador

Fluxo Principal de Eventos

1. Na tela do jogo, o jogador solta uma bomba;
2. A bomba explode e atinge uma parede, que ao ser destruída mostra o item fogo que estava escondido;
3. O jogador move-se em direção ao item, e ao passar por ele, aumenta o alcance de sua bomba;
4. O caso de uso é finalizado.

Fluxo de Exceção

Não há.

Regras de Negócio

Não há.

Caso de Uso 008 – Pegar item skate

Descrição

Este caso de uso serve para aumentar a velocidade do personagem do jogador.

Pré – condições

No início do jogo, o item skate deve estar “escondido” em uma “parede”.

Pós – condições

Não há.

Ator(es)

Jogador

Fluxo Principal de Eventos

1. Na tela do jogo, o jogador solta uma bomba;
2. A bomba explode e atinge uma parede, que ao ser destruída mostra o item skate que estava escondido;
3. O jogador move-se em direção ao item, e ao passar por ele, sua velocidade fica maior;
4. O caso de uso é finalizado.

Fluxo de Exceção

Não há.

Regras de Negócio

Não há.

Caso de Uso 009 – Pegar item bomba

Descrição

Este caso de uso serve para que o jogador possa soltar mais de uma bomba ao mesmo tempo.

Pré – condições

No início do jogo, o item bomba deve estar “escondido” em uma “parede”.

Pós – condições

Não há.

Ator(es)

Jogador

Fluxo Principal de Eventos

1. Na tela do jogo, o jogador solta uma bomba;
2. A bomba explode e atinge uma parede, que ao ser destruída mostra o item bomba que estava escondido;
3. O jogador move-se em direção ao item, e ao passar por ele, poderá soltar duas bombas seguidas;
4. O caso de uso é finalizado.

Fluxo de Exceção

Não há.

Regras de Negócio

R1 O número de itens bomba que o jogador conseguir pegar, corresponde ao número de bombas que ele poderá soltar ao mesmo tempo.

Caso de Uso 010 – Consultar pontuação

Descrição

Este caso de uso serve para que o jogador consulte sua pontuação segundo suas partidas online e também sua posição no ranking.

Pré – condições

O jogador deve estar logado.

Pós – condições

Não há.

Ator(es)

Jogador

Fluxo Principal de Eventos

1. Na tela inicial, o jogador seleciona a opção “Pontuação”.
2. O sistema mostra uma tela com a pontuação e a posição do jogador no ranking.
3. O caso de uso é finalizado.

Fluxo de Exceção

Não há.

Regras de Negócio

R1 A opção “Pontuação” ficará bloqueada se o jogador não estiver logado.

Caso de Uso 011 – Consultar conquistas

Descrição

Este caso de uso serve para que o jogador consulte suas conquistas segundo suas partidas online.

Pré – condições

O jogador deve estar logado.

Pós – condições

Não há.

Ator(es)

Jogador

Fluxo Principal de Eventos

1. Na tela inicial, o jogador seleciona a opção “Conquistas”.
2. O sistema mostra uma tela com as conquistas do jogador.
3. O caso de uso é finalizado.

Fluxo de Exceção

Não há.

Regras de Negócio

R1 A opção “Conquistas” ficará bloqueada se o jogador não estiver logado.

Caso de Uso 012 – Gerenciar dados de relatórios

Descrição

Este caso de uso serve para que o administrador gerencie os dados das partidas dos jogadores e monte os relatórios para a consulta.

Pré – condições

Pós – condições

Não há.

Ator(es)

Administrador

Fluxo Principal de Eventos

1. O administrador acessa sistema de relatórios;
2. O administrador seleciona o tipo de relatório desejado;
3. O sistema mostra opções de filtragem;
4. O administrador informa os dados para a consulta e clica em “Buscar”;
5. Sistema filtra e mostra os dados;
6. Sistema abre uma tela com uma tabela;

7. O administrador insere os dados na tabela mostrada pelo sistema;
8. O administrador salva os dados;
9. O caso de uso é finalizado.

Fluxo de Exceção

E1 Nenhum dado encontrado para o relatório escolhido

E1.1 Sistema retorna mensagem “Nenhum dado encontrado”.

Regras de Negócio

Não há.

APÊNDICE C – DIAGRAMA ENTIDADE RELACIONAMENTO

DICIONÁRIO DE DADOS

ENTIDADE: users
- Contém as informações do jogador;
- Relaciona-se com a tabela **highscore**;

Atributos:
usr_id: ID do usuário (Primary Key)
username: Nome do usuário
passwrd: Senha do usuário
email: E-mail do usuário

ENTIDADE: highscore
- Contém informações dos outros possíveis highscores;

Atributos:
users_usr_id: ID do usuário (Foreign Key)
highscoreBattlestadium_id: ID do highscore battlestadium (Foreign Key)
scoreGame_id: ID do scoreGame (Foreign Key)

ENTIDADE: scoreGame
- Contém as informações dos jogos disputados pelo usuário;
- Realciona-se com a tabela **highscore;**

Atributos:
id: ID da pontuação (Primary Key)
enemiesKilled: Número de inimigos derrotados
wallsDestroyed: Número de paredes destruídas
itemsUsed: Número de itens usados
timeGame: Tempo jogado

ENTIDADE: highscoreBattlestadium
- Contém as informações de pontuação do módulo BattleStadium;
- Realciona-se com as tabelas **highscore** e **highscoreSingle;**

Atributos:
id: ID do highscoreBattlestadium (Primary Key)
tipoHighScore: Tipo de pontuação

ENTIDADE: highscoreSingle
- Contém as informações das partidas SingleMatch;
- Relaciona-se com as tabelas **highscoreBattlestadium** e **highscoreItems**;

Atributos:
idhighscoreSingle: ID do highscoreSingle
highscoreBattlestadium_id: ID do highscoreBattlestadium (Foreign Key)
highscoreItems_idhighscoreItems: ID do highscoreItems (Foreign Key)

ENTIDADE: highscoreItem
- Contém as informações do contém as informações dos Itens que compõem o HighScoreSingle;
- Relaciona-se com a tabela **highscoreSingle**;

Atributos:
idhighscoreItems: ID do highscoreItems (Primary Key)
vitoria: Número de vitórias do jogador
derrota: Número de derrotas do jogador
trofeu: Quantidade de troféus do jogador
bobersaldo: Saldo do jogador

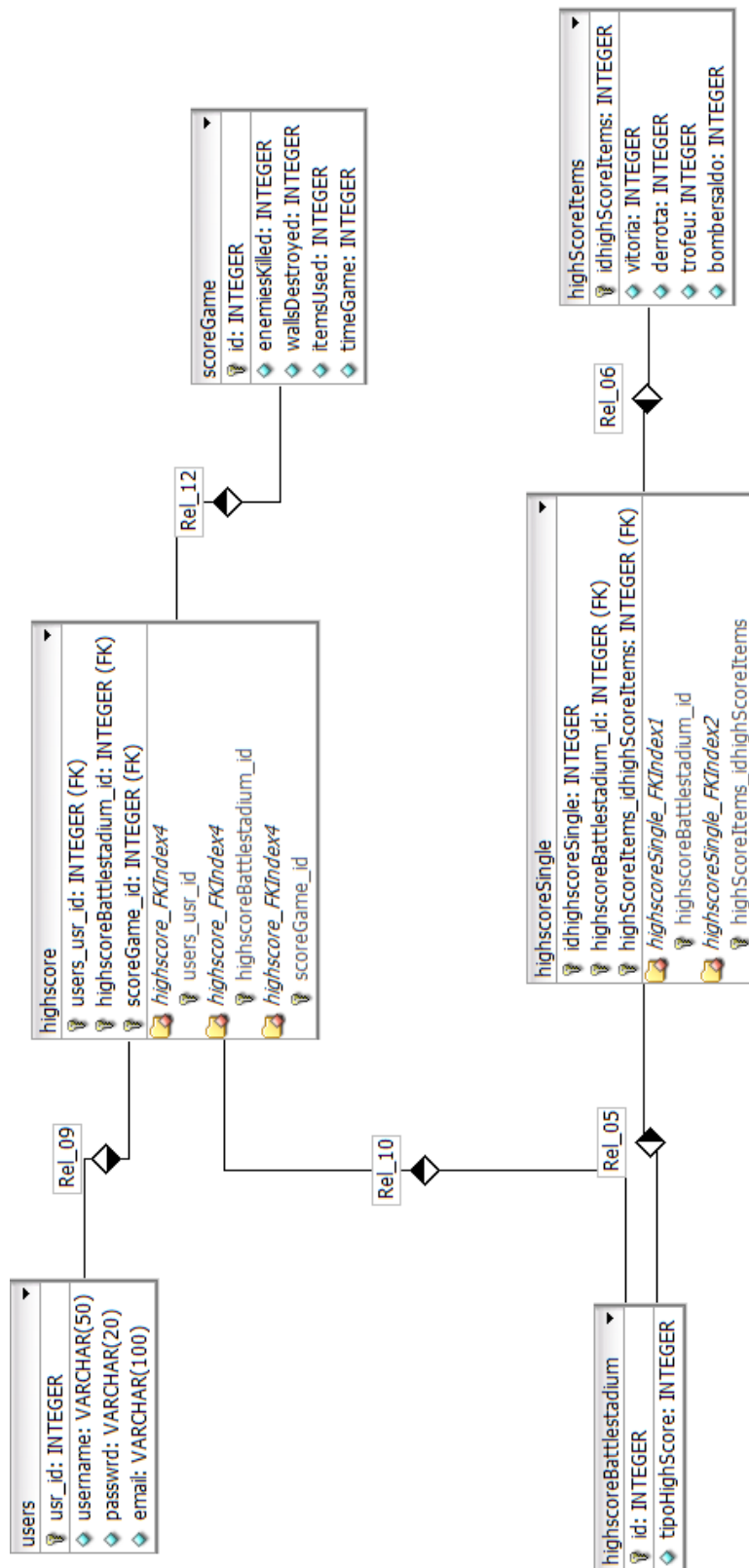



FIGURA – Diagrama Entidade Relacionamento

APÊNDICE D – DIAGRAMA DE CLASSES

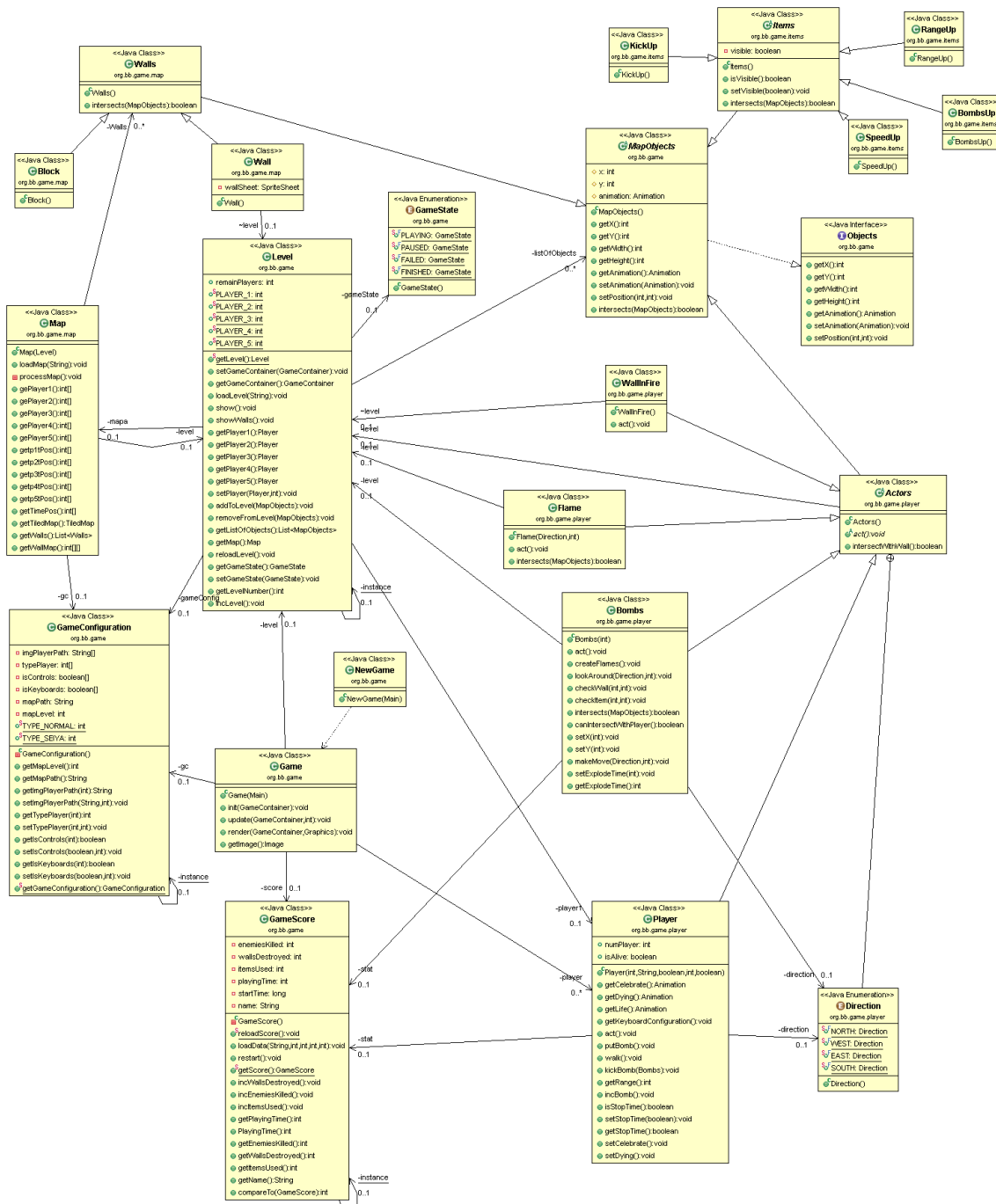



FIGURA – Diagrama de Classes – Módulo Offline

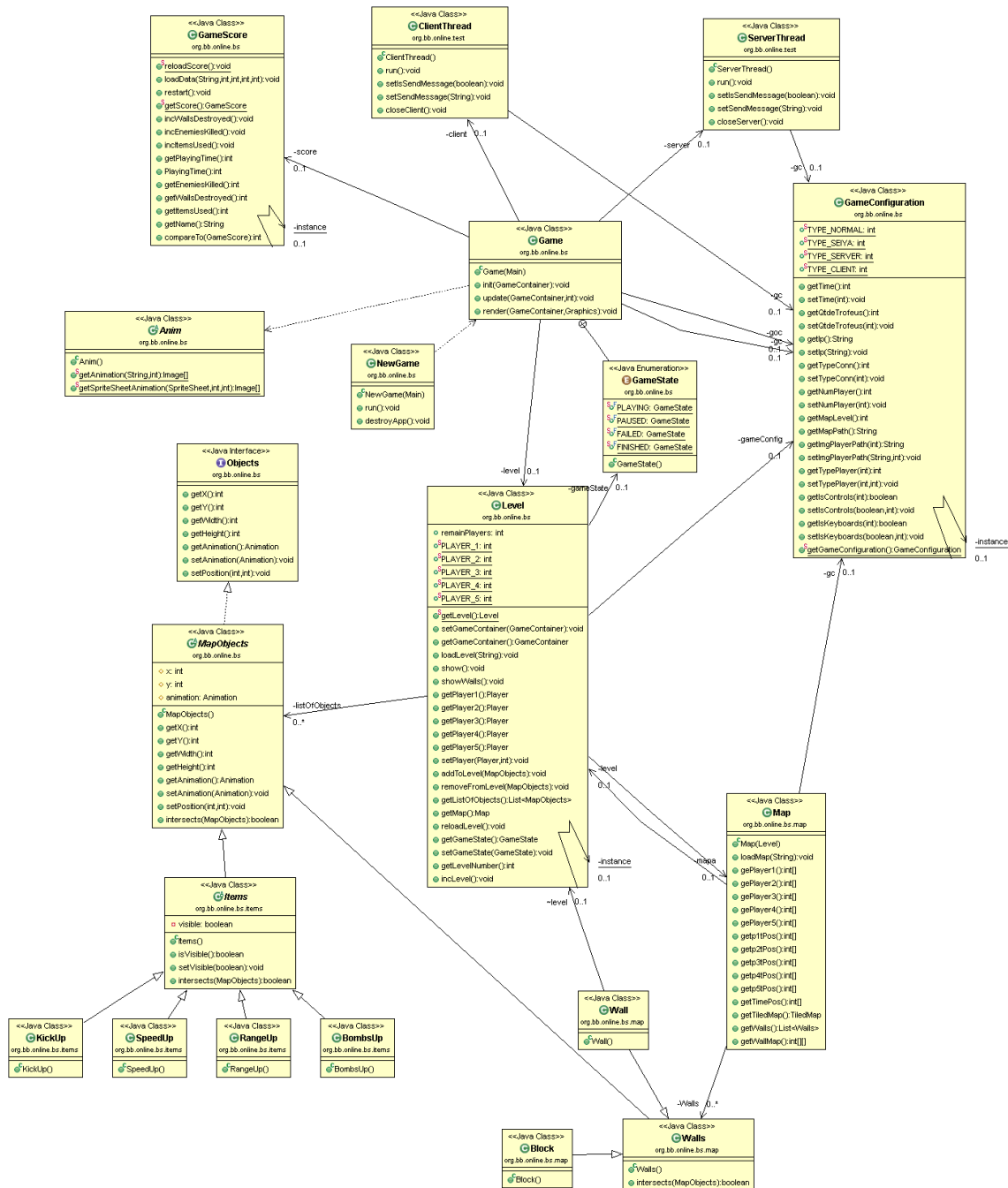



FIGURA – Diagrama de Classes – Módulo Online

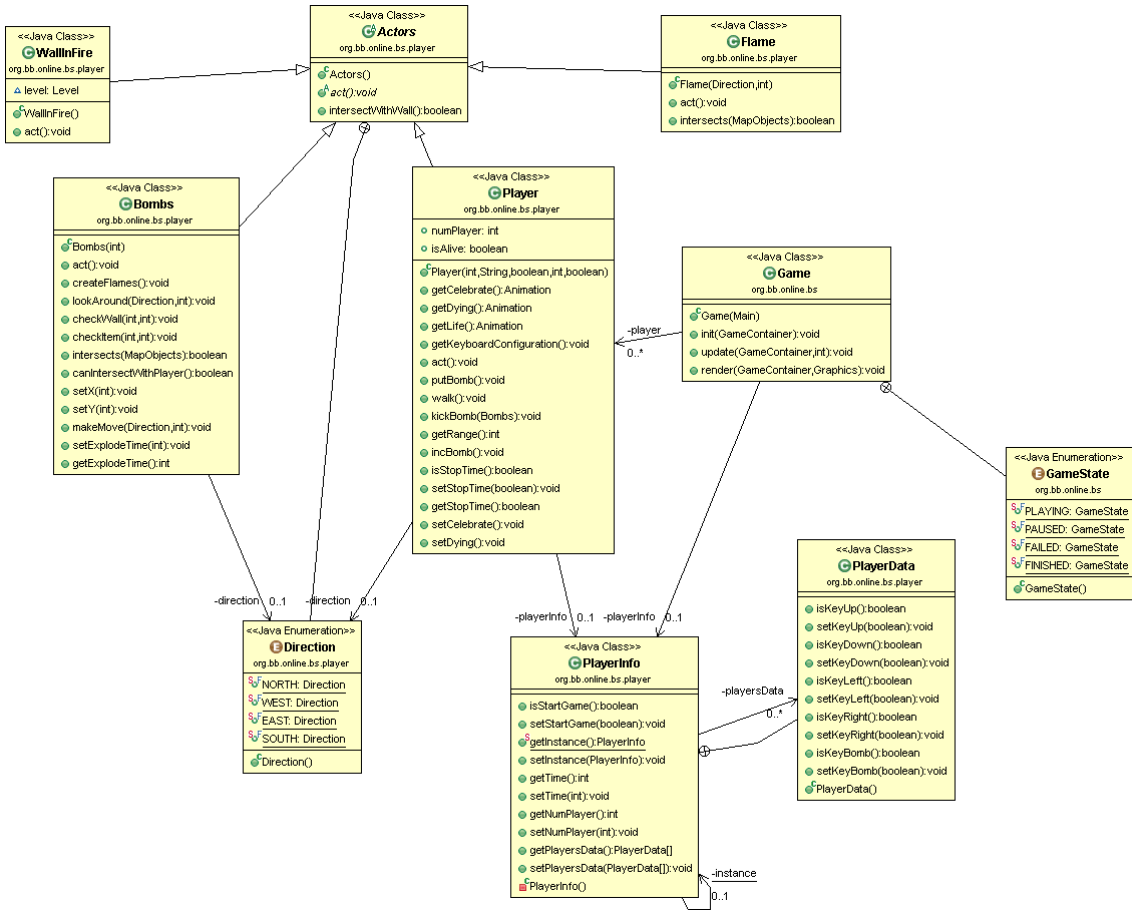



FIGURA – Diagrama de Classes – Módulo Online – Player

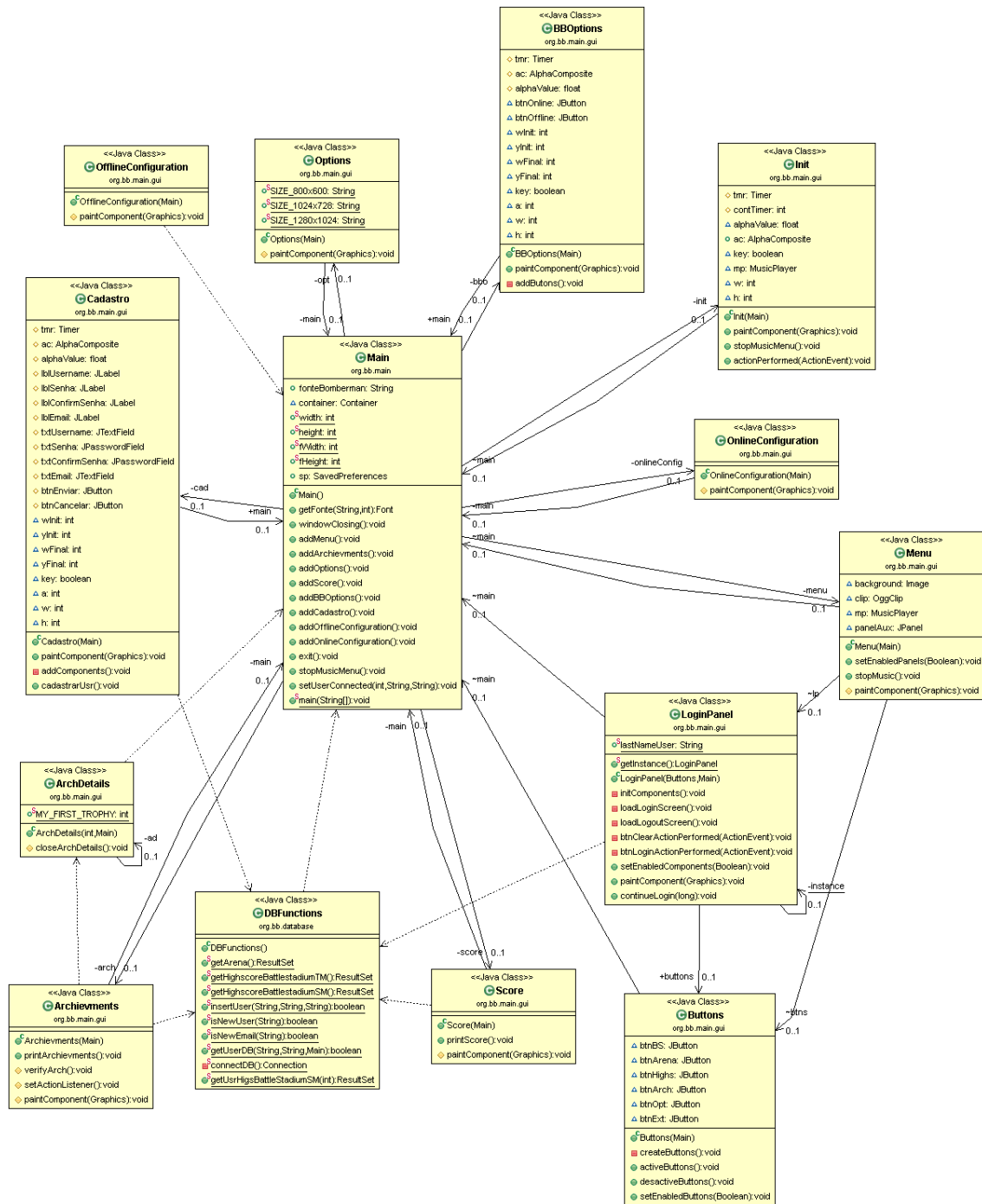



FIGURA – Diagrama de Classes – Interface

APÊNDICE E – DIAGRAMA DE SEQUÊNCIA

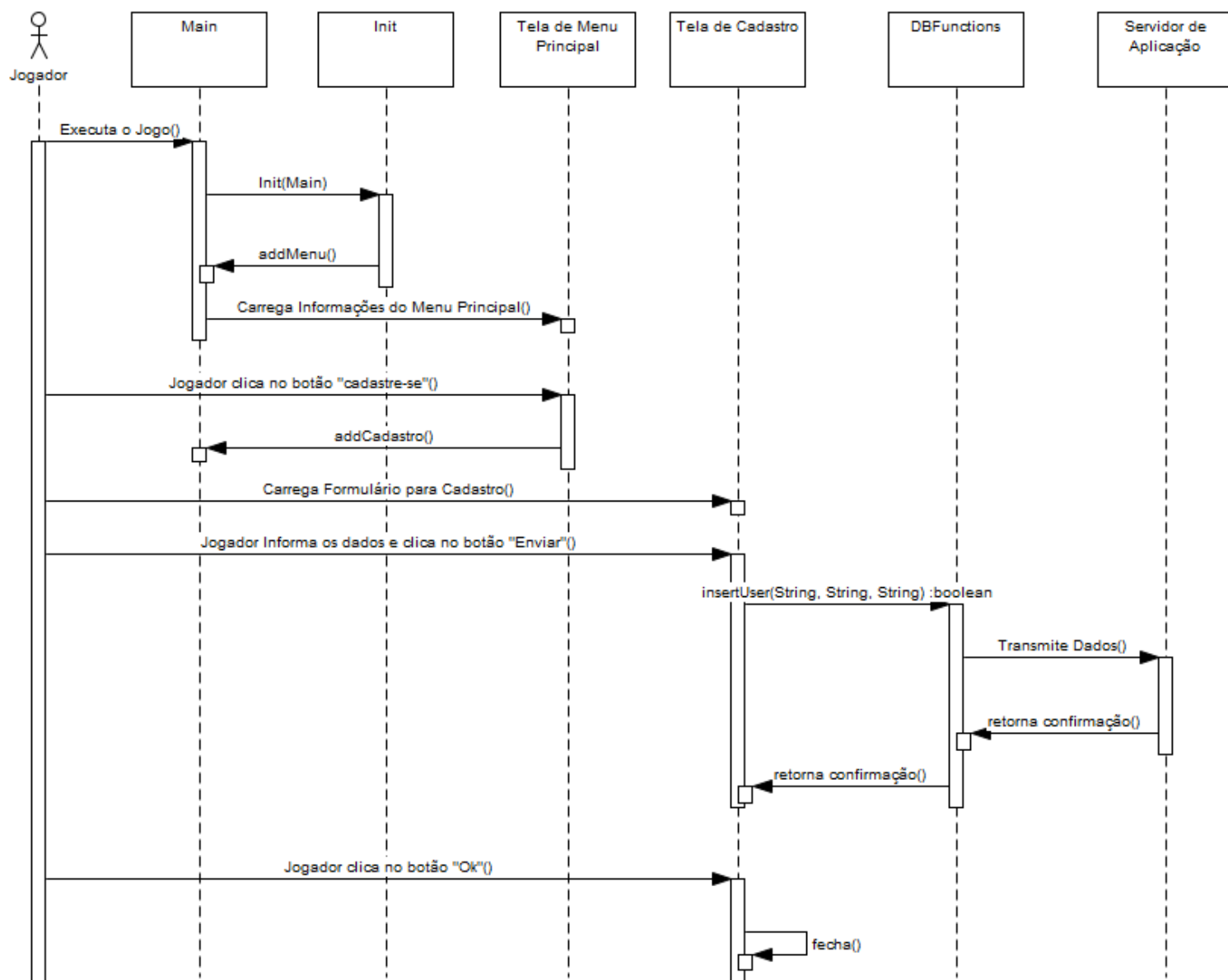



FIGURA – Diagrama de Sequência – Início do Jogo e Cadastro de Usuário

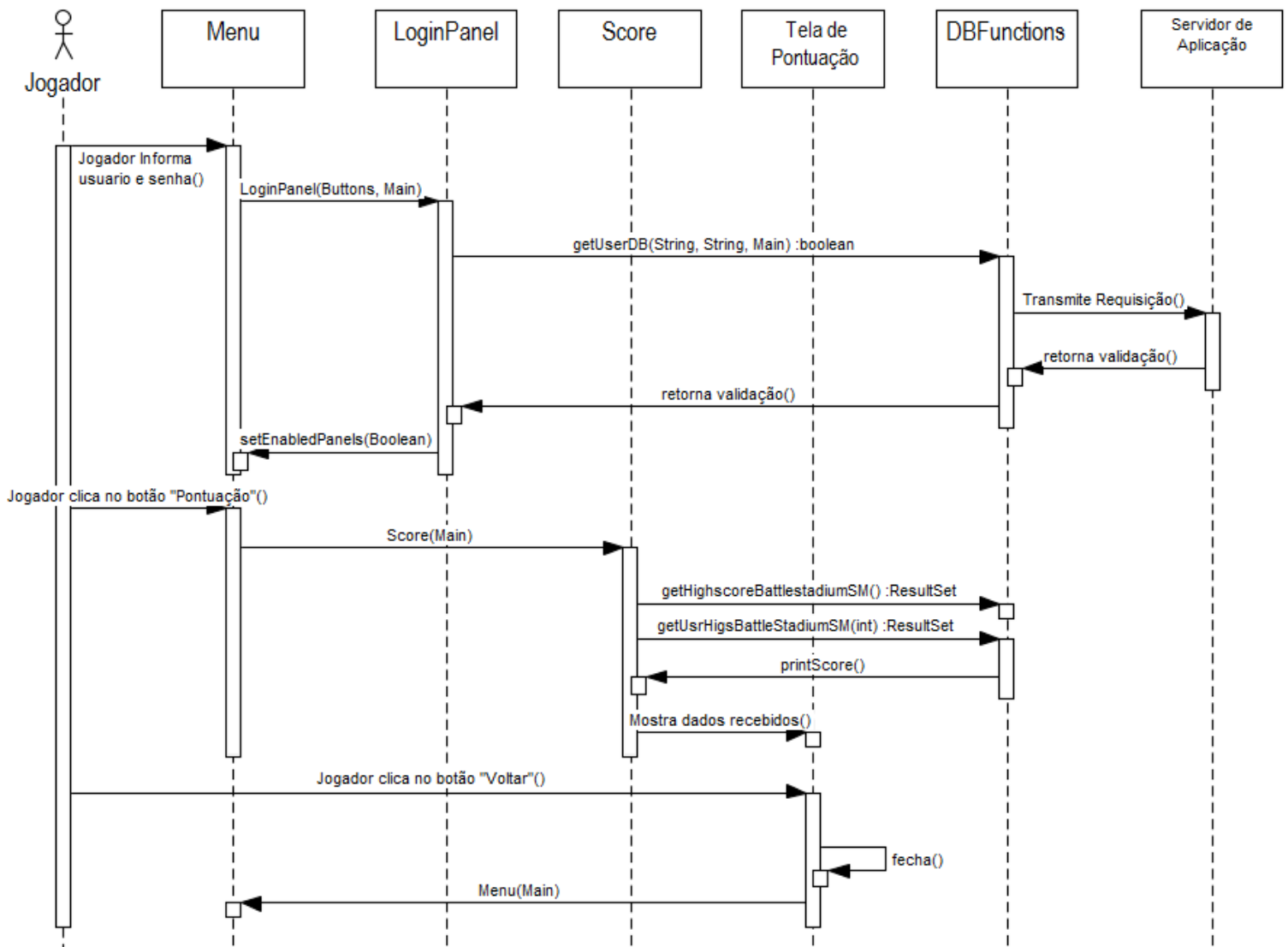



FIGURA – Diagrama de Sequência – Login e Pontuação

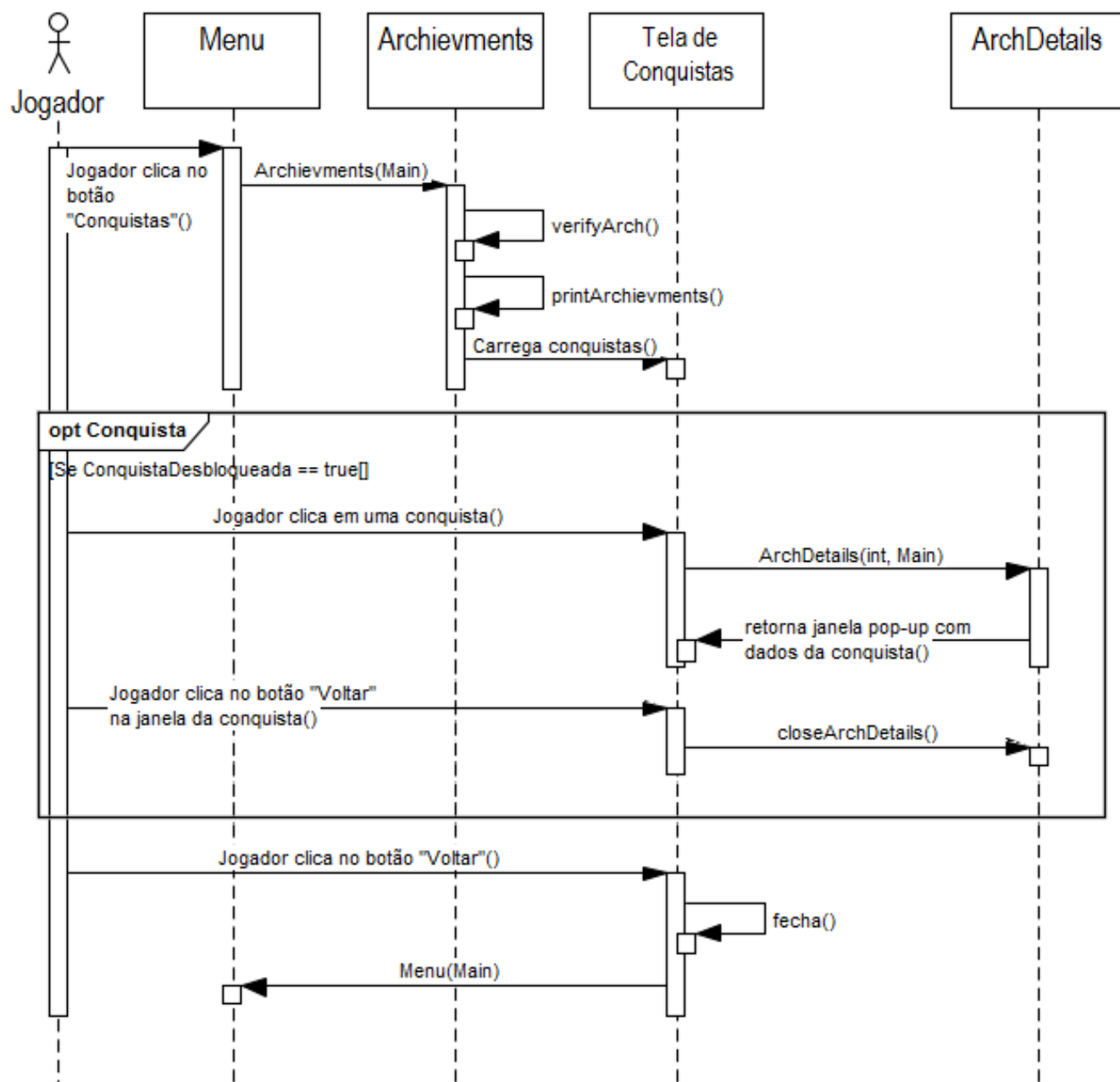



FIGURA – Diagrama de Sequência – Conquistas

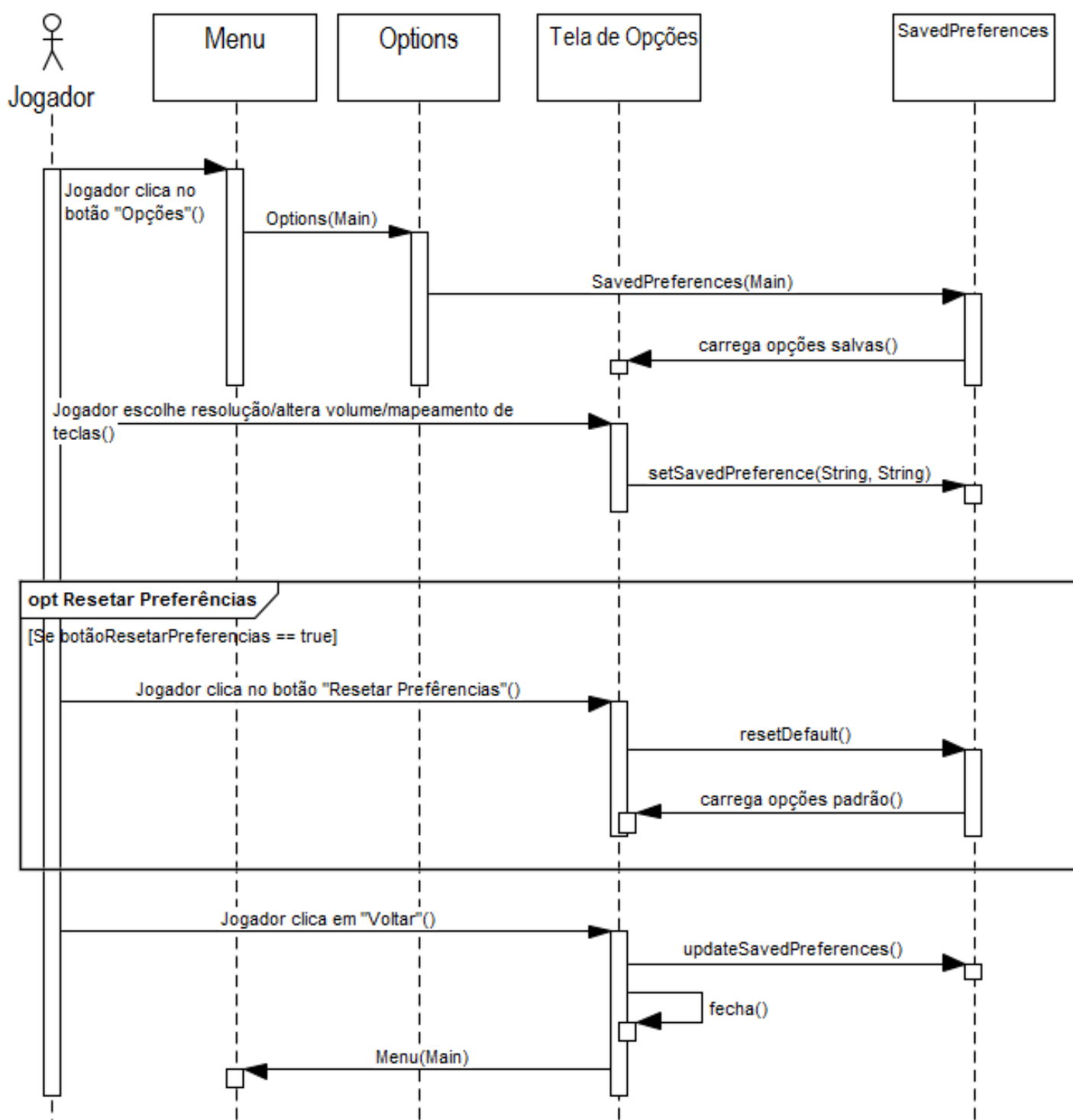



FIGURA – Diagrama de Sequência – Opções do jogo

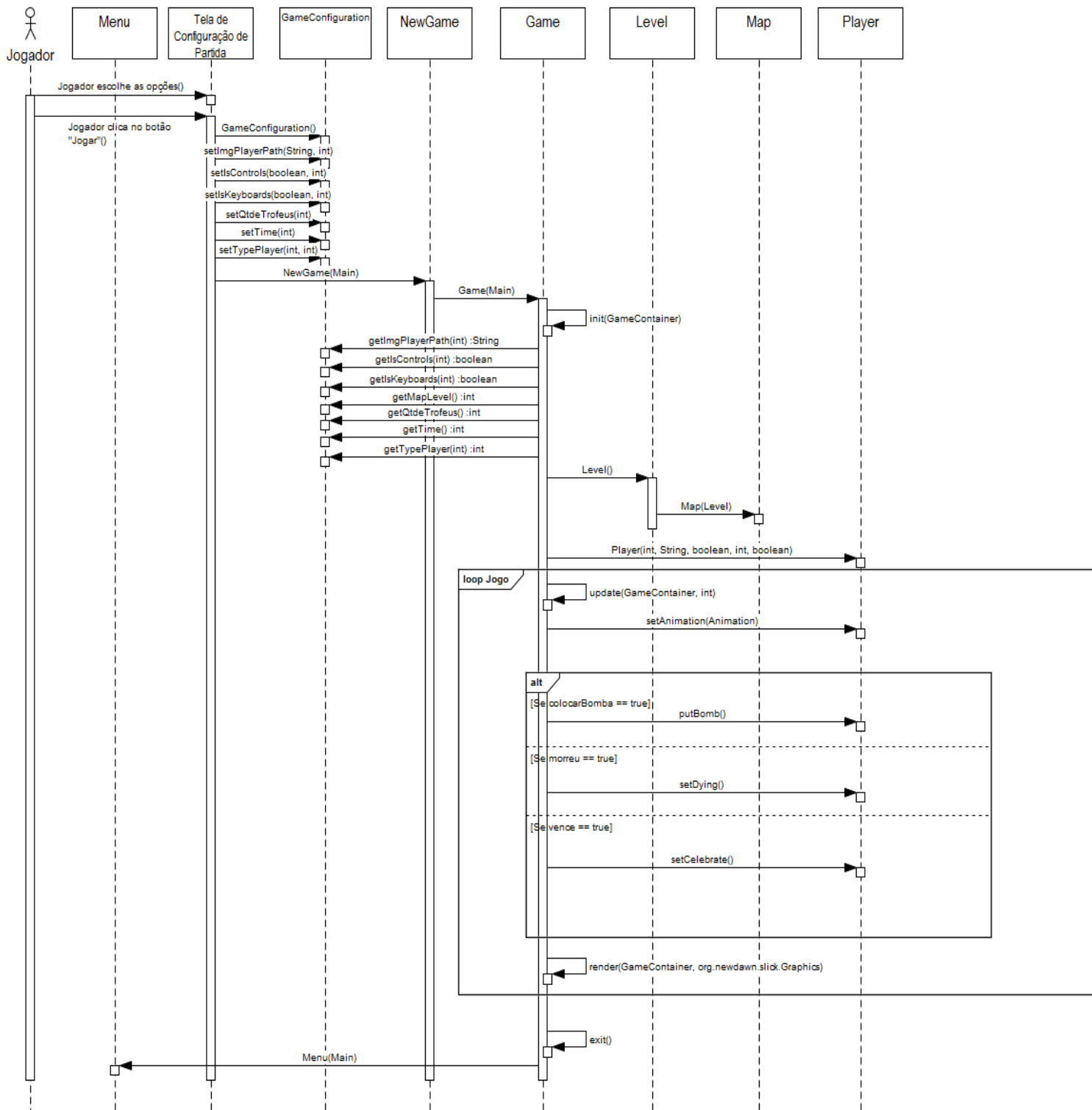



FIGURA – Diagrama de Sequência – Configuração e Jogo BattleStadium Offline

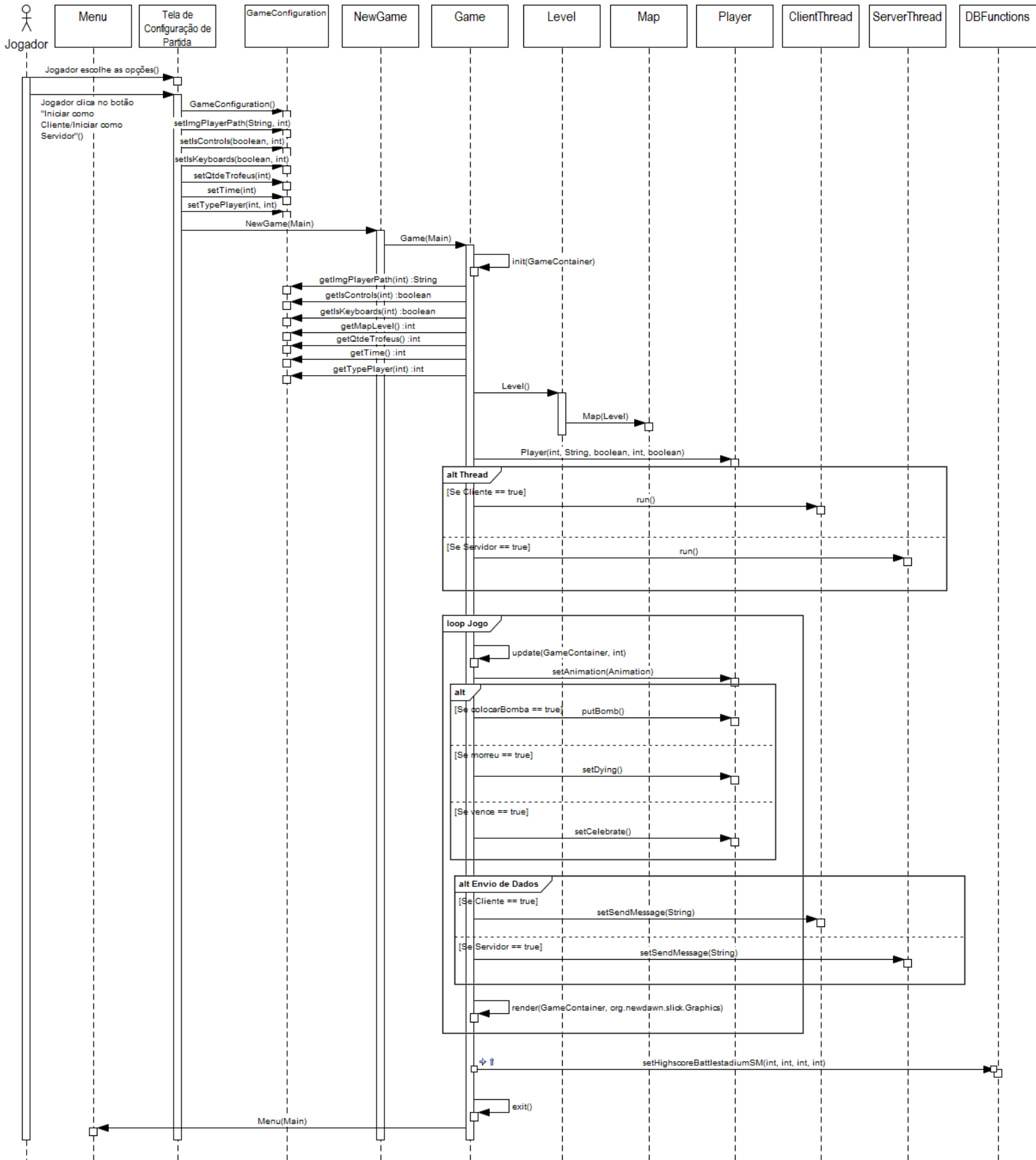



FIGURA – Diagrama de Sequência – Jogo BattleStadium em Rede Local